Approximation hochdimensionaler komplexer Prozessdaten in der Metallurgie

D i s s e r t a t i o n

zur Erlangung des Doktorgrades

der Ingenieurwissenschaften

vorgelegt von

Marc Winter

aus Saarbrücken

genehmigt von der Fakultät für Natur- und Materialwissenschaften

der Technischen Universität Clausthal

Tag der mündlichen Prüfung

25. 4. 2014

Dekan

Prof. Dr. Winfried Daum

Vorsitzender der Promotionskommission

Prof. Dr.-Ing. Heinz Palkowski

Betreuer

Prof. Dr.-Ing. Karl-Heinz Spitzer

Gutachter

Prof. Dr.-Ing. Karl-Hermann Tacke

Kurzfassung

Diese Arbeit befasst sich mit dem Problem, funktionale Zusammenhänge zwischen gemessenen Materialeigenschaften und denjenigen Parametern herzustellen, die den zugrundeliegenden Herstellungsprozess charakterisieren. Hierzu wurden verschiedene Datensätze aus der Grobblechherstellung untersucht. Eine besondere Schwierigkeit besteht darin, eine Vielzahl von oftmals gegenläufigen Produktanforderungen zu realisieren und gleichzeitig das Risiko für fehlerhafte Produkte zu minimieren. Die Frage nach einem vorhersagefähigen funktionalen Zusammenhang zwischen Prozess und Produkt ist deshalb von fundamentaler Bedeutung für die gezielte Weiterentwicklung und Verbesserung neuer Produkte im Bezug auf die gewünschten Materialeigenschaften. Die in dieser Arbeit vorgestellten Verfahren können prinzipiell auf beliebige auf kontinuierlichen Parametern basierenden Datensätze angewendet werden.

Die Approximation erfolgt empirisch, indem automatisiert geeignete Anpassungsfunktionen generiert werden, die darauf abzielen, ein Maximum an Information aus den existierenden Messdaten zu extrahieren. Hierzu wurden zwei Algorithmen entwickelt, die auf Polynomen sowie neuronalen Netzen basieren. Die Algorithmen generieren ausgehend von einfachen Basismodellen sukzessive Modelle mit steigender Komplexität, bis eine gute Approximations- und Generalisierungsfähigkeit erreicht wurde. Die Algorithmen zielen dabei auf hochdimensionale Probleme mit verhältnismäßig wenigen Messdaten ab und berücksichtigen explizit die Abbildung schwellwertbehafteter Daten, um eine bessere Anpassungsfähigkeit zu erzielen.

Die Modelle wurden sowohl untereinander, als auch mit Standardverfahren verglichen. Dazu kamen neben den realen Datensätzen auch künstlich generierte Datensätze zum Einsatz, die in realen Daten auftrete Probleme simulieren sollen. Auf allen Datensätzen konnte die Leistungsfähigkeit der Algorithmen demonstriert werden.

An einigen praktischen Beispielfällen werden Einsatzmöglichkeiten der Modelle aufgezeigt, die im realen Einsatz bereits geholfen haben, neue Produkte zu entwickeln und zu verbessern.

Abstract

This work focuses on the problem of finding functional relationships between material properties and the process parameters describing the underlying generating process. In this context, several datasets from heavy plate production have been analysed. A particular difficulty arises from accomplishing in many cases opposing product requirements and simultaneously minimising the risk of obtaining faulty products. Thus the demand for a predictive functional relationship between process and product is of fundamental importance for a purposeful development and improvement of new products with respect to the demanded material properties. The methods presented in this work can in principle be applied to arbitrary data sets based on continuous parameters.

The approximation is performed in an empirical manner by automatically generating fitting functions which aim at extracting a maximum amount of information from the existing measurement data. To approach this task, two algorithms have been developed based on polynomials and neural networks. Starting from simple basic models, these algorithms successively build up models of increasing complexity until a good approximation and generalisation capability is achieved. The algorithms focus on high-dimensional problems with relatively few measurements and explicitely consider the reproduction of thresholded data to achieve a better level of approximation.

The models were compared to each other and to standard methods. This was done both on real data and artificially generated datasets simulating problems known from real data. The performance of the algorithms was demonstraded on all data sets.

Potential applications of the models were demonstrated on some practical examples which already helped in developing and improving new products.

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Problemstellung und Motivation

Das Ergebnis naturwissenschaftlicher Prozessese hängt oft von einer Vielzahl Parameter ab, über deren Wirkung im Detail nur wenig Information vorliegt. Dabei kann es sich beispielsweise um physikalische Phänomene, technische Abläufe oder chemische Reaktionen handeln. Ein Ansatz zum Erlangen von Erkenntnissen zur Kontrolle eines Prozesses besteht in der sukzessiven Untersuchung der Wirkung der einzelnen den Prozess beschreibenden Parameter und derern Wechselwirkungen untereinander auf das Ergebnis („Bottom-up"). Ein anderer Ansatz besteht in der statistischen Auswertung des Ergebnisses als Funktion der Prozessparameter („Top-down"). Der Bottom-up Ansatz ist bei komplexen Systemen, d. h. solchen mit vielen Parametern und komplizierten Zusammenhängen, meistens mit einer hohen Anzahl Versuche verbunden, die sich nicht mit vertretbarem Aufwand durchführen lassen. In solchen Fällen kann statt dessen versucht werden, ein Vorhersagemodell für das Prozessergebnis aus einer begrenzten Menge von Versuchsergebnissen mithilfe eines Top-down Ansatzes zu erstellen.

In dieser Arbeit werden die Ergebnisse verschiedener Herstellungsprozesse von Stahlblechen nach Top-down Ansätzen beschrieben. Die Eigenschaften eines fertigen Blechs werden durch die chemische Zusammensetzung und die Behandlungsschritte beim Durchlaufen der Prozesskette vom Gießen des Stahlstrangs über das Walzen zum Blech bis hin zum Kühlen bestimmt.

Die für die Produkteigenschaften relevante Prozesskette beginnt mit dem Frischen des Roheisens zum Stahl. Dabei wird Sauerstoff eingeblasen, um den Kohlenstoffgehalt zu verringern und weitere unerwünschte Begleitelemente wie Phosphor auf ein Minimum zu reduzieren. Anschließend wird der Stahl durch gezielte Zugabe von Legierungselementen wie Mangan, Niob oder Titan legiert und homogenisiert, um eine vorgegebene chemische Zusammensetzung zu erzielen [1]. Durch eine Vakuumbehandlung zur Entgasung wird der Reinheitsgrad erhöht.

Der flüssige Stahl wird danach zu Brammen vergossen, die in mehreren Schritten zu Blechen gewalzt werden. Die während und nach der Erstarrung auftretenden Abkühl- und Aufheizvorgänge laufen mit Geschwindigkeiten ab, die keine Beschreibung des Systems durch thermodynamische Gleichgewichtszustände erlauben. Statt dessen werden komplexe Nichtgleichgewichtszustände durchlaufen, deren Beschreibung durch Bottom-up Ansätze in vielen Fällen sehr aufwändig und mangels Detailinformationen nicht immer möglich ist.

Durch die chemische Zusammensetzung und die Gießbedingungen wird während der Erstarrung das Wachstum der Dendriden [2] und die Entstehung von Seigerungsgebieten [3] gesteuert. Der Grad der Verformungen von der Bramme zum Blech und die dabei herrschenden Temperaturen, Abkühlgeschwindigkeiten und Wartezeiten in den Zwischenschritten bestimmt die Art und die Umwandlung der Polykristalle sowie deren Wechselwirkungen mit mikroskopischen Ausscheidungen [4] und Einschlüssen [5]. Durch die gezielte Anwendung thermo-mechanischer Walzverfahren [6] und Kühlprozesse [7] sollen bestimmte Produkteigenschaften wie beispielsweise Festigkeit, Zähigkeit und Korrosionsbeständigkeit erzielt werden.

Aus dieser Produktionskette ergeben sich die Prozessparameter im Herstellungsprozess, welche die Qualität des Produkts bestimmen [8, 9]. Beispielsweise kann die Festigkeit des Stahls erhöht werden, indem die Größe der Kristallite verringert wird [10, 11]. Dies wird durch gezielte thermo-mechanische Umformung während des Walzens erreicht. Durch den Einsatz geringer Mengen von Mikrolegierungselementen wie Al, Nb, Ti oder V, die während der Umformung als Nitride oder Carbide ausscheiden, kann das Kornwachstum gehemmt werden, wodurch die Kornfeinung verstärkt wird. Dadurch erhöht sich gemäß der Hall-Petch-Beziehung

$$R_e = s_0 + \frac{K}{\sqrt{d_K}} \tag{1.1.1}$$

die Festigkeit des Werkstoffs [12]. Hierbei bezeichnet s_0 die Spannung, ab der die Versetzungsbewegung in einem großen Kristalliten beginnen würde, K den Korngrenzenwiderstand und d_K den mittleren Durchmesser der Körner.

Die Kornfeinung wirkt sich ebenfalls positiv auf die Zähigeit aus [11]. Eine feine Korngröße kann weiterhin durch beschleunigte Abkühlung nach dem thermo-mechanischen Umformen erzielt werden. Temperaturen und Umformgrade beim thermo-mechanischen Walzen, Legierungselementkonzentrationen und Abkühlgeschwindigkeiten sind Prozessparameter zur Steuerung der Gefügebildung. Deren Wirkung auf bestimmte davon abhängige Produktmerkmale wurde in der Literatur bereits in vielen Spezialfällen ausführlich untersucht. Dabei wurden Bottom-Up Ansätze basierend auf physikalisch-chemischen Modellierungen [13] sowie experimentelle Methoden angewendet. Diese Erkenntnisse können aber oft nicht direkt auf die Produktionsbedingungen übertragen werden, da Produktionsstähle meist komplexer

sind als im Labor realisierte Modellsysteme und die Wirkungen einzelner Prozessparameter sich gegenseitig beeinflussen können. Bei einigen der untersuchten Eigenschaften können die gefundenen Abhängigkeiten mit ähnlichen Fällen aus der Literatur verglichen werden, während für andere Eigenschaften lediglich qualitative Erkenntnisse vorliegen.

Moderne Stähle müssen für spezielle Anwendungsbereiche optimiert werden. Die geforderten Materialeigenschaften lassen sich im Allgemeinen aber nicht unabhängig voneinander einstellen. Die Variation eines oder mehrerer Prozessparameter zur Verbesserung einer bestimmten Eigenschaft des Endprodukts kann eine Verschlechterung anderer Eigenschaften zur Folge haben. Ein Beispiel hierfür ist die Erhöhung des Kohlenstoffgehalts im Stahl, um die Streckgrenze zu erhöhen. Der höhere Kohlenstoffanteil hat eine Verringerung der Zähigkeit zur Folge, was in vielen Fällen unerwünscht ist. Quantitative Vorhersagen der Eigenschaftsveränderungen sind zur Optimierung der Prozessvorgaben unabdingbar. Um das Wissen über die Zusammenhänge zwischen Herstellungsprozess und Produkteigenschaften effizient weiterzuentwickeln, sind gezielte Verbesserungsschritte nötig. Dazu können Vorhersagemodelle verwendet werden, die Produkteigenschaften auch für Kombinationen von Prozessparametern abschätzen können, die bislang noch nicht messtechnisch erfasst worden sind.

Diese Arbeit hat zum Ziel, Zusammenhänge zwischen Prozessparametern und bestimmten Eigenschaften des Produkts abzubilden. Dazu werden Messdaten aus der Produktion mit statistischen Modellen ausgewertet. Der Schwerpunkt liegt auf der Auswertung von Problemen, bei denen verhältnismäßig wenig Messdaten vorliegen im Vergleich zur Anzahl der Prozessparameter, was besondere Ansprüche an die Fähigkeiten der Modelle stellt. Durch die Gegenüberstellung verschiedener Modelltypen soll ein Vergleich der erhaltenen Erkenntnisse sowie eine Auswahl des für die jeweilige Problemstellung am besten geeigneten Modells erzielt werden.

Die daraus gewonnenen Erkenntnisse können die Entwicklung zukünftiger Stähle entweder durch die Vorhersagen der Modelle für vorgegebene Prozessparameter unterstützen oder selbst verbesserte Prozessparameter vorschlagen. Hierzu können numerische Optimierer verwendet werden, die basierend auf den Modellwerten eine Verbesserungsrichtung bestimmen [14]. Diese Optimierung kann zusätzlich an Nebenbedingungen geknüpft werden. Dadurch kann beispielsweise eine Produkteigenschaft verbessert werden unter der Bedingung, dass bestimmte Produktparameter oder andere Produkteigenschaften innerhalb vorgegebener Grenzen bleiben. Diese können ebenfalls durch Modelle vorhergesagt oder die direkt vorgegeben werden, z. B. Kostengrenzen als Funktion der Legierungselementkosten.

1.2 Komplexe Systeme

Die grundlegenden Prozesse bei der Entstehung des Stahls spielen sich auf atomarer Ebene und auf Kristallgitterebene ab. Die Bestandteile des Systems durchlaufen dabei eine komplexe Reaktionskinetik unterschiedlicher kristalliner Phasen. Zu Beginn der Erstarrung existieren bereits Fluktuationen in der Temperaturverteilung und chemischen Zusammensetzung der Schmelze. Diese werden zwar prozesstechnisch so gering wie möglich gehalten, lassen sich aber nicht vollständig vermeiden. Das Wachstum der einzelnen Kristalle und deren Umwandlung in verschiedene Kristalltypen während der folgenden Abkühl- und Umformschritte hängt von der Wechselwirkung mit deren Umgebung ab. Durch die inter- und intrakristallinen Reaktionen und Stoffdiffusion generiert das System ständig neue lokale Fluktuationen der chemischen Zusammensetzung (Seigerungen [15]), die den Ausgangszustand für die darauf folgenden Wachstums- und Umwandlungsschritte bildet. Hierdurch können benachbarte Kristalle auf unterschiedlichen Zeit- und Entfernungsskalen miteinander wechselwirken. Hinzu kommen Interaktionen mit nichtkristallinen Phasen, die ebenso das Kristallwachstum und die Materialeigenschaften der fertig erstarrten Phase beeinflussen. In den hierbei beschriebenen Prozessen treten nichtlineare Wechselwirkugen auf, durch die die lokale zeitliche Entwicklung auf mikroskopischer Ebene sensibel sowohl gegenüber lokalen Fluktuationen in den Anfangsbedingungen, als auch großräumigen Schwankungen während des Entstehungsprozesses wird. Diese Eigenschaften komplexer Systeme führen dazu, dass selbst unter identischen Prozessvorgaben einzelne Bleche aus der selben Prozesskette unterschiedliche Eigenschaften aufweisen können. Dieser Effekt wirkt sich auf manche Produkteigenschaften stark aus, auf andere nur geringfügig. Modellierungen des Gesamtsystems basierend auf first principles gestalten sich aufgrund der Komplexität als nicht praktikabel.

Viele in der modernen Wissenschaft untersuchte Systeme zeigen komplexes Verhalten, etwa in der Physik, Chemie, Biologie, Medizin, Meteorologie, Ökonomie und Soziologie [16]. Auch in den Ingenieurwissenschaften hat sich die Beschreibung der Physik komplexer Systeme mittlerweile zu einer wichtigen Disziplin entwickelt [17, 18]. In der Stahlentwicklung finden sich immer mehr komplexe Systeme, z. B. in einem Problem beim Strangguss, das hier betrachtet wird [19, 20, 21, 22], die bei der Vorhersage von Produktionsergebnissen unbedingt berücksichtigt werden müssen, da sonst fehlerhafte Folgerungen aus Produktionsergebnissen oder Versuchen gezogen werden können.

1.3 Stand des Wissens

Es existiert eine Vielzahl statistischer Methoden, die Zusammenhänge zwischen einer abhängigen Variable und mehrerer unabhängiger Variablen suchen [23]. Jede dieser Methoden macht Annahmen zum Verhalten des abzubildenden Zusammenhangs und bietet Vor- und Nachteile, die bei verschiedenen Pro-

blemen unterschiedlich stark ins Gewicht fallen. Welches Modell sich für ein bestimmtes Problem am besten eignet, hängt von der Komplexität des Problems sowie der Menge und Qualität der vorhandenen Daten ab. Um einen Modellierungsansatz überhaupt rechtfertigen zu können, wird im Allgemeinen die Annahme gemacht, dass ein kausaler Zusammenhang zwischen den Prozessparametern und den Eigenschaften des Stahls existiert. Die gemessenen Daten sind zusätzlich mit zufälligen Fehlern behaftet.

Zur Modellierung der Probleme kommen in dieser Arbeit zwei unterschiedliche inter- und extrapolationsfähige statistische Ansätze zum Einsatz, die mit einer rein klassifizierenden Methode verglichen werden. Die skalaren Prozessparameter stellen dabei die unabhängigen Variablen eines Modells dar. Sie werden als Eingangsgrößen bezeichnet. Aus diesen wird die abhängige Variable, die Zielgröße, als Ausgabe des Modells berechnet.

1.3.1 Methoden für kontinuierliche Zusammenhänge

Hierbei wird versucht, die Zielgröße durch einen kontinuierlichen funktionalen Zusammenhang abzubilden. Dies geschieht beispielsweise bei der Regressionsanalyse oder bei künstlichen neuronalen Netzen.

Die Regressionsanalyse (Abschnitt 1.5) passt eine vorgegebene mathematische Funktion derart an, dass die Fehler zwischen Funktionswerten und zugehörigen Messwerten bezüglich einer vorgegebenen Norm minimal sind. Falls physikalisches Vorwissen über den Funktionsverlauf existiert, kann dieses durch Vorgabe der Funktion explizit eingebracht werden, z. B. Exponentialfunktionen zur Beschreibung der Änderung der Konzentration bei chemischen Reaktionen. Falls kein Vorwissen über die gesuchte Funktion existiert, können die Messdaten durch einfache Modellfunktionen approximiert werden, z. B. lineare Funktionen oder Polynome höherer Ordnung. Wenn die anzupassenden Regressionsparameter des Modells selbst nichtlinear in die Funktionsgleichung eingehen, stehen Methoden der nichtlinearen Regression zur Verfügung [24, 25]. Probleme, bei denen sie in linearer Form eingehen, können mit linearen Regressionsmethoden gelöst werden, siehe Abschnitt 7.1. Die Wirkungen der Prozessparameter selbst auf die Zielgröße dürfen dabei durchaus nichtlinear sein. Die Regression kann global erfolgen, indem die Funktion an alle Messdaten angepasst wird oder lokal, indem um den zu modellierenden Punkt eine Regression erfolgt, die entfernte Punkte weniger stark gewichtet. Diese ursprünglich eindimensionale lokale Regression [26] lässt sich auf beliebig viele Dimensionen erweitern, wenn ein mehrdimensionales Abstandsmaß definiert wird. Die lokale Anpassung kann daher komplexe Funktionen im Allgemeinen genauer abbilden als eine globale Anpassung von gleichem Grad, allerdings auf Kosten der Verallgemeinerungsfähigkeit. Sie wird hauptsächlich zur Glättung verrauschter Daten und Interpolation verwendet [27].

Künstliche Neuronale Netze (Abschnitt 7.2) sind an die Funktionsweise des Gehirns angelehnt und bilden diese stark vereinfacht nach, indem die Eingangsgrößen ein Netz aus Aktivierungsfunktionen

durchlaufen. Um das Netz an die Messdaten anzupassen, ist es in der Praxis üblich, die einzelnen Netzverbindungen unterschiedlich stark zu gewichten, so dass die Zielfunktion möglichst genau abgebildet wird [28]. Diese Modelle bieten durch die Charakteristik ihrer Aktivierungsfunktionen den Vorteil, auch komplexe und unstetige Zusammenhänge gut abbilden zu können [29]. Dadurch können auch Daten analysiert werden, die sich nicht explizit durch funktionale Zusammenhänge beschreiben lassen, z. B. Mustererkennung in der Bildverarbeitung.

1.3.2 Methoden für klassifizierte Zusammenhänge

Methoden zur Beschreibung klassifizierter Zusammenhänge teilen die Eingangsgrößen in Klassen ein und ordnen jeder Klasse einen Wert der Zielfunktion zu. Beispiele hierfür sind Entscheidungsbäume oder Bayessche Netze [30]. Entscheidungsbäume sind eine grafische Darstellung der Daten derart, dass für jede Eingangsgröße mehrere Klassen gebildet werden, die nacheinander mit weiteren Klassen verknüpft werden, bis die gesamte Menge der Eingangsdaten abgedeckt ist. Bei einem Bayesschen Netz repräsentieren die Verbindungen bedingte Wahrscheinlichkeiten [30]. Bayessche Netze werden in dieser Arbeit zum Verlgeich mit den Methoden für kontinuierliche Zusammenhänge herangezogen.

Um eine Modellfunktion in kleinen Schritten um einen Ausgangspunkt herum auf Veränderungen einer Zielgröße hin untersuchen zu können, muss diese Funktion wenigstens in diesem Bereich in der Lage sein, die Daten durch eine stetige Funktion wiederzugeben. Wenn diese Wiedergabe innerhalb des durch die Messdaten definierten Bereichs geschehen soll, spricht mach von Interpolation, andernfalls von Extrapolation. Weiterhin sollte sie in der Lage sein, Veränderungen der Zielgröße auch bei kleiner Variation von Parametern abbilden zu können. Klassifizierende Methoden sind dazu nicht in der Lage, da sie nur eine Aussage innerhalb der durch die Klassifizierung definierten Gebiete machen können und jeder Klasse einen konstanten Modellwert zuordnen. Durch eine Erhöhung der Anzahl der Klassen lässt sich eine kontinuierliche Verteilung zwar immer feiner approximieren, allerdings fallen dann immer weniger Messdaten in die einzelnen Klassen. Dadurch wird das Modell weniger stabil gegenüber Streuungen der Messwerte. Bei einer zu hohen Anzahl an Intervallen lernt das Modell nur „auswendig” und verliert die Eigenschaft zu verallgemeinern.

Da eine reine Klassifizierung im Gegensatz zur kontinuierlichen Modellierung nicht durch die zusätzliche Bedingung eingeschränkt wird, stetig inter- und extrapolieren zu können, kann sie als Vergleich herangezogen werden, wie exakt die Messungen eines Datensatzes bestenfalls wiedergegeben werden können. Eine 100%ige Wiedergabe ist nicht möglich, wenn mehrere unterschiedliche Messungen zu gleichen Eingangsdaten vorliegen, was aufgrund von Messstreuungen passieren kann.

1.4 Datenerfassung

Um ein beliebiges Modell sinnvoll trainieren zu können, müssen die Ausgangsdaten in einer geeigneten Form vorliegen und plausibel sein. Die Erfassung und zweckmäßige Aufbereitung der Daten ist aus verschiedenen Gründen nicht trivial.

1.4.1 Verfügbarkeit repräsentativer Daten

Es kann nicht immer garantiert werden, dass alle zur Beschreibung des Problems signifikanten Informationen in passender Form verfügbar sind. Die für einen Teilprozess charakteristischen und im Sinne der Produktion sinnvoll messbaren Parameter reichen nicht immer aus, um einen komplexen funktionalen Zusammenhang einer Zielgröße beliebig genau zu beschreiben. So können beispielsweise während des Walzens lediglich mittlere Oberflächentemperaturen gemessen werden. Detaillierte Messungen des Temperaturverlaufs im Inneren des Blechs, der die Gefügestruktur beeinflusst, sind aber nicht möglich und können bestenfalls durch Temperaturmodelle approximiert werden. Weierhin kann sich die Bedeutung bestimmter Parameter ändern, wenn Prozessabläufe mit der Zeit geändert bzw. erweitert werden. Diese Umstände schränken unabhängig vom Modell die Aussagekraft der zur Beschreibung des Problems gewählten Produktionsdaten ein.

1.4.2 Auswahl eines repräsentativen Datensatzes

Bei den in dieser Arbeit untersuchten Aufgabenstellungen ist a priori nicht für alle zur Verfügung stehenden Prozessparameter genau bekannt, ob sie einen signifikanten Einfluss auf das Ergebnis haben. Die probeweise Einbeziehung einer hohen Anzahl an Parametern erhöht allerdings den zeitlichen Aufwand der Modellerstellung und macht die Modelle fehleranfälliger. Die Ermittlung eines repräsentativen Parametersatzes ist in erster Linie dem Anwender überlassen, kann aber auch als Teil der Modellerstellung betrachtet werden.

Um eine Korrelation zwischen Eingangsgrößen und Zielgröße überhaupt erst mit statistischen Methoden rekonstruieren zu können, darf die Anzahl der Eingangsgrößen nicht zu hoch sein gegenüber der Anzahl der Messergebnisse. Deshalb müssen einzelne Prozessschritte, die durch eine zu hohe Anzahl von Prozessparametern beschrieben werden, durch einen kleineren Satz möglichst repräsentativer Parameter dargestellt werden. Dies beschränkt die Fähigkeit eines Modells, beliebig detaillierte Zusammenhänge zwischen Eingangsgrößen und Zielgröße erlernen zu können. Beispielsweise wird die Gefügestruktur eines Blechs gegebener chemischer Zusammensetzung durch den Prozess des Walzens der Bramme zur Walztafel und des Kühlens derselben nach dem Walzprozess bestimmt. Das Walzen erfolgt dabei in mehreren Schritten, sog. Stichen, bei dem die Dicke jeweils um einen vorgegebenen Faktor und mit einer

bestimmten Geschwindigkeit reduziert wird. Die Anzahl der Stiche kann durchaus im Bereich 10-20 oder mehr liegen. Würde man Umformgrad, -geschwindigkeit und Temperatur bei jedem Stich sowie die Verweilzeit zwischen den einzelnen Stichen ins Modell aufnehmen, entstünde eine sehr hohe Anzahl an Parametern. Da die Umformung gegen Ende der Walzphase aber üblicherweise einen größeren Einfluss auf die Gefügestruktur hat als diejenige zu Beginn, kann eine entsprechende Wahl einiger weniger Parameter gegen Ende der Walzphase zusammen mit dem Gesamtumformgrad die Walzung ausreichend genau charakterisieren.

1.4.3 Rauschen und Fluktuationen

Bei realen Messungen können zufällige Fehler auftreten, die das Messergebnis mehr oder weniger stark verfälschen. Weiterhin sind die Materialeigenschaften innerhalb des Volumens der Probe aufgrund lokaler Variationen während des Prozesses nicht konstant. Diese nicht erfassten Einflussgrößen führen ebenfalls zu Streuungen der Messwerte. Die in Versuchen ermittelten Messgrößen können stark von mikroskopischen Unregelmäßigkeiten der Probe abhängen, so z. B. beim Kerbschlagbiegeversuch [31] oder BDWT-Test [32], bei denen die Widerstandsfähigkeit des Materials gegen Rissausbreitung, charakterisiert duch dessen Zähigkeit [33], bestimmt wird. Die Messergebnisse variieren mit der Konzentration potentieller Rissentstehungszentren und der Gebiete, in denen der Riss aufgrund eines zu hohen Rissausbreitungswiderstands gestoppt wird. Da für eine Probe meistens nur einzelne oder wenige Messungen vorliegen, lässt sich die Verteilung der Messwerte in der Regel nicht direkt überprüfen. Es wird angenommen, dass diese Fehler symmetrisch um den wahren Wert herum verteilt sind, so dass der Mittelwert der gemessenen Größe mit steigender Anzahl Wiederholungsmessungen immer genauer dem statistischen Mittel entspricht.

Es ist nicht immer möglich, sämtliche für die Entstehung der Zielgröße relevanten Prozessparameter explizit zu berücksichtigen. Messungen zu gleichen Vektoren von Eingangsparametern können deshalb unterschiedliche Messergebnisse liefern, was zu einer weiteren Streuung führt, die durch ein Modell nicht eliminiert werden kann. Unabhängig von der Wahl des Modells ist eine exakte Wiedergabe aller Messdaten in solchen Fällen nicht möglich.

Unter dem Begriff „Rauschen” werden im Folgenden alle Phänomene zusammengefasst, die dazu führen, dass wiederholte Messungen zum gleichen Satz von Prozessparametern unterschiedliche Ergebnisse liefern.

1.4.4 Fehlerhafte Daten

Bei den Prozessparametern handelt es sich um Soll-Vorgaben oder gemessene Werte, die ebenfalls vom tatsächlichen Wert abweichen können. Variationen der Zielgröße, die aufgrund von nicht oder ungenau gemessenen Parametern sowie weiteren Unregelmäßigkeiten entstehen, können nicht von zufälligen Messfehlern unterschieden werden und werden daher im Folgenden ebenfalls unter dem Begriff „Rauschen" mitgeführt.

Weiterhin können Daten falsch eingegeben oder aufgrund von Prozessstörungen ausgelassen werden. Falsch eingegebene Daten lassen sich oft durch Plausibilitätsprüfungen identifizieren und entfernen. In diesem Fall muss allerdings der gesamte Datenpunkt entfernt werden, da die meisten Modelle nur bei vollständiger Angabe eines Parametersatzes arbeiten können. Andernfalls kann keine eindeutige funktionale Beziehung zur Zielgröße hergestellt werden. Im Fall von Entscheidungsbäumen ist es allerdings möglich, unvollständige Datensätze auf Kosten erhöhter Komplexität der Baumstruktur explizit einzubinden und dadurch zumindest einen Teil der Information zu nutzen [30]. Bayes-Netze können ebenfalls mit unvollständigen Eingangsdaten arbeiten (Abschnitt 7.3).

Fehlerhafte Daten führen zu Außreißern im Modell, da sie nicht dem Verlauf der zugrundeliegenden funktionalen Abhängigkeit folgen. Sie sind für die Entwicklung von Modellen störend, da sie die den restlichen Daten zugrunde liegende Information verzerren und daher nicht in die Berechnung des Modells eingehen sollten. Vermutliche Ausreißer können in einfachen Fällen durch Auftragung der Zielgröße gegen die Eingangsparameter entdeckt werden oder es können die Messwerte betrachtet werden, die stark von einer Modellvorhersage abweichen [34]. Dies allein ist aber kein sicheres Kriterium für das Finden von Ausreißern. Der Ansatz ist nur dann sinnvoll, wenn in der Umgebung genügend weitere Messwerte vorhanden sind, die sich mit den Modellwerten gut decken. Solche Untersuchungen machen daher nur dann Sinn, wenn niedrigdimensionale Datensätze mit verhältnismäßig vielen Messungen untersucht werden, was nicht Ziel dieser Arbeit ist. Bei hochdimensionalen, dünn besetzten Daten besteht die Gefahr, dass Daten, die nicht ins Modell passen, voreilig als Ausreißer entfernt werden, obwohl der Grund für die schlechte Anpassung auch ein unzureichendens Modell sein könnte.

1.4.5 Schwierigkeiten aufgrund der Beschaffenheit der Datensätze

Durch Vorgaben und Einschränkungen bei der Auftragsplanung sind Messreihen, bei denen lediglich ein einziger Parameter variiert wird, während alle anderen konstant bleiben, praktisch unmöglich. Statt dessen variiert bei unterschiedlichen Produktionen üblicherweise ein ganzer Satz von Prozessparametern mehr oder weniger stark. Das bedeutet zwar für ein multivariantes statistisches Modell keine Einschränkung, macht aber eine anschauliche Auftragung der Zielgröße gegen einen bestimmten Parameter

zur direkten Kontrolle unmöglich. Selbst Datenpunkte, bei denen die Parameter jeweils nur relativ gering voneinander abweichen, können bei komplexen Systemen zu deutlich unterschiedlichen Ergebnissen führen.

Die für verschiedene Aufträge erstellten Produktklassen führen zu einer Clusterung der Eingangsparameter. Ändert sich ein Eingangsparameter innerhalb eines solchen Clusters nur wenig, dann fällt der Messfehler zwischen einzelnen Punkten verhältnismäßig stark ins Gewicht. Daher ist es für eine Auswertung nicht nur von Bedeutung, wie viele Messungen bei einer gegebenen Zahl von Parametern zur Verfügung stehen und wie stark die Messwerte streuen, sondern auch wie gleichmäßig die Messungen den Parameterraum abdecken.

1.4.6 Beispiele für Clusterung

Abbildung 1.1 zeigt ein Beispiel anhand der durch die Funktion $f(x_i) = x_i + \varepsilon$ generierten Funktionswerte an unterschiedlich verteilten Stützstellen x_i, wobei dem wahren zugrundeliegenden Zusammenhang $y = x_i$ ein Rauschen durch Zufallszahlen ε aus dem Intervall $[-0,2; 0,2]$ überlagert wurde. Ein Fall simuliert eine Clusterung durch Wahl der Eingangsparameter x_i innerhalb eines schmalen Intervalls $0 < x_i < 0.05$. Im anderen Fall sind die x_i bei gleicher Anzahl an Datenpunkten über das Intervall $0 < x_i < 1$ verteilt. Die lineare Anpassung an die in Clustern angeordneten Daten wird verhältnismäßig stark durch das Rauschen gestört und gibt die Funktion $f(x) = x$ schlechter wieder als die lineare Anpassung an die über das Intervall $0 < x < 1$ generierten Daten.

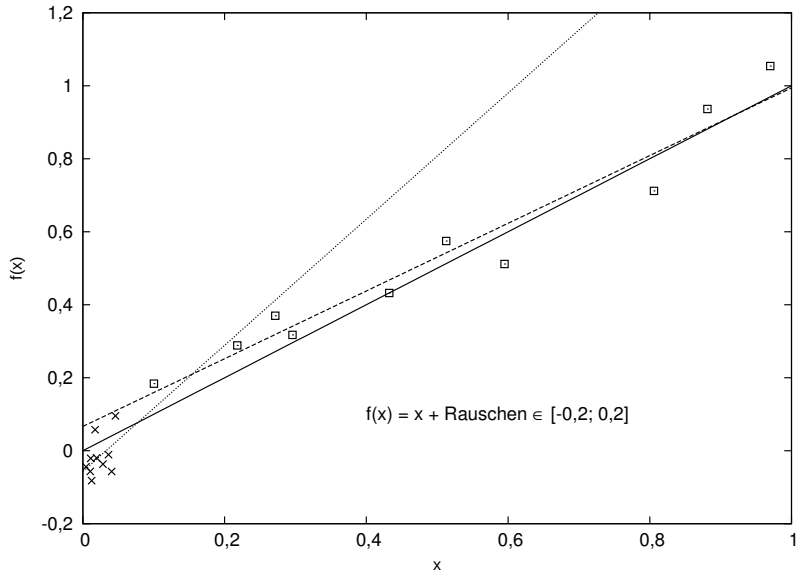


Abbildung 1.1: *Die beiden Klassen von Messpunkten stellen die gleiche mit konstantem Fehler überlagerte Funktion y (durchgezogene Linie) über unterschiedlich große Intervalle dar. Die gestrichelten Linien zeigen die entsprechenden linearen Anpassungen. Im Fall des stark beschränkten Eingabeintervalls (gepunktete Linie) verschlechtert das Rauschen die Anpassung stärker als im breiteren Eingabeintervall (gestrichelte Linie).*

Abbildung 1.2 a) zeigt die Clusterung von Produktionsdaten bezüglich der Legierungselemente Cu und Ni für eine Reihe von Auftragsklassen. Wenn die Zielgröße von weiteren Eingangsgrößen abhängt, zerfällt die Verteilung im höherdimensionalen Raum in immer mehr Cluster. Ein Modell extrapoliert dann bezüglich eines Parameters unter Umständen bereits in einem leeren bzw. dünn besetzten Bereich zwischen Clustern, obwohl alle einzelnen Parameter noch innerhalb der Messintervalle $[x_{min} \leq x_i \leq x_{max}]$ liegen. Da Extrapolation gerade bei verrauschten Daten schnell zu unrealistischen Vorhersagen führen kann, ist bei den Vorhersagewerten eines Modells auch zu prüfen, ob der betrachtete Punkt in einer Umgebung liegt, die mit Produktionsdaten abgedeckt ist. Hierzu muss im Parameterraum zunächst ein angemessenes Abstandsmaß für die Datenpunkte definiert werden. Dessen Wahl ist zunächst willkürlich, da die Parameter im Allgemeinen unterschiedliche physikalische Einheiten haben. In hochdimensionalen Räumen wie in dieser Arbeit betrachtet kann die Identifizierung, ob ein Punkt „nah" an einem anderen liegt, problematisch sein [35]. „Nah" im dem Sinne, dass eine Variation eines Parameters nur einen geringen Einfluss auf die Zielgröße hat, lässt sich nur dann definieren, wenn die Zielgröße bereits bekannt ist.

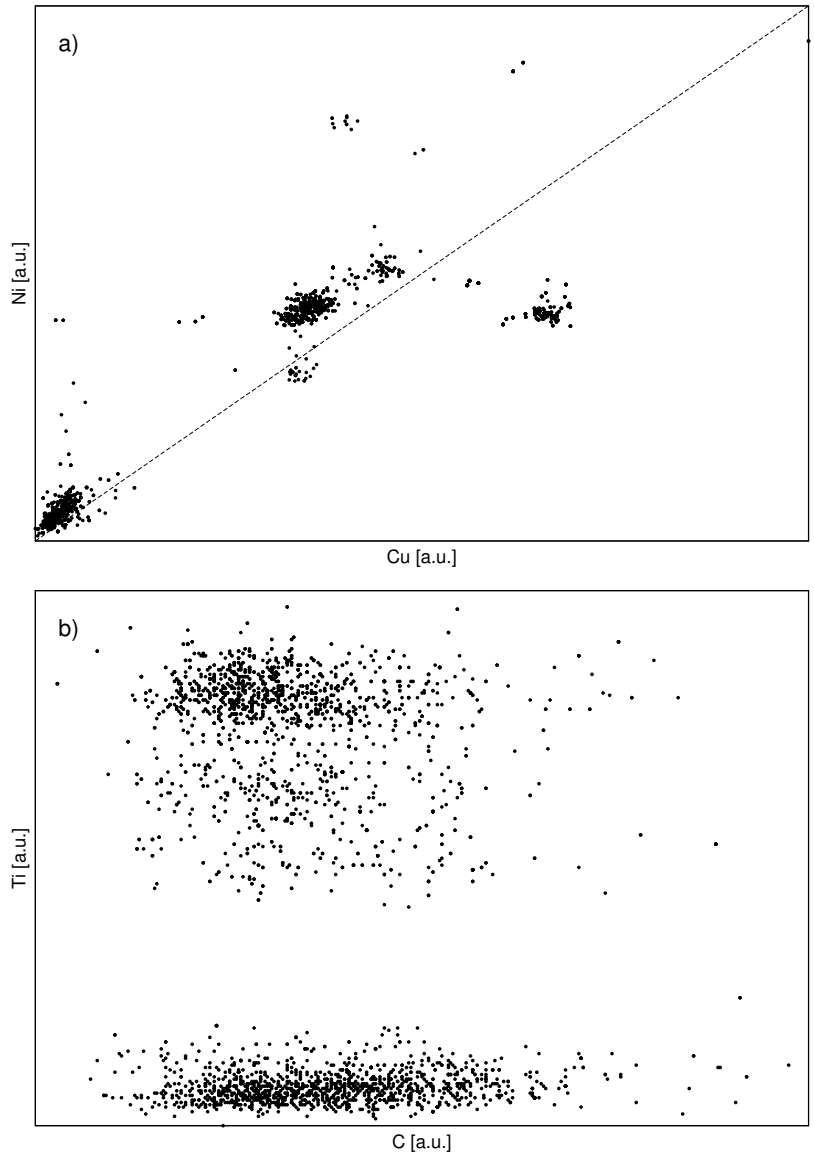


Abbildung 1.2: *Verteilung verschiedener Legierungselemente: a)* Cu *und* Ni *stark korreliert, b)* C *und* Ti *nicht korreliert.*

Viele Datenpunkte in Abbildung 1.2 a) liegen dicht an einer Ursprungsgeraden, d. h. dass die beiden Parameter Cu und Ni stark miteinander korreliert sind. Bei einer solchen Korrelation ist es kaum noch möglich, die Wirkung der einzelnen Elemente auf die Zielgröße voneinander zu unterscheiden. Die Information über die unterschiedlichen Einflüsse steckt dann in den wenigen Punkten, die weiter von der Korrelationsgeraden entfernt liegen. Abbildung 1.2 b) zeigt die Verteilung von C und Ti. Hier besteht keine Korrelation und die Wirkungen der beiden Elemente können klar voneinander getrennt werden.

Wenn die Zielgröße von vielen Eingangsparametern abhängt und der Datenraum entsprechend hoch-

dimensional ist, sind exponentiell viele Datenpunkte nötig, um genügend Information zur Modellerstellung zur Verfügung zu stellen. Setzt man zur approximativen Beschreibung einer Zielgröße für jeden der m Eingangsparameter je N Messpunkte voraus, dann sind zur Abdeckung des gesamten Datenraums N^m Datenpunkte notwendig. Selbst für eine grobe Betrachtung mit nur je einem Messpunkt an der Untergrenze, in der Mitte und an der Obergrenze des zu betrachtenden Intervalls $(N = 3)$ ergeben sich beispielsweise für $m = 10$ bereits 59.049 Datenpunkte, für $m = 15$ etwa $1,4 \cdot 10^7$ und für $m = 20 \approx 3 \cdot 10^9$. Dieser „Fluch der Dimensionalität" [36] macht es in der Praxis oft unrentabel oder unmöglich, den gesamten Datenraum mit einer entsprechend hohen Anzahl Messdaten zu beproben. Daher sollte bei der Anwendung der Modelle darauf geachtet werden, dass Vorhersagen nicht in Bereichen gemacht werden, die unzureichend von Messdaten gestützt werden. In praktischen Anwendungen ist aber oft nicht nach einer zuverlässigen Vorhersage innerhalb des gesamten Datenraums gefragt, sondern nur im Bereich einiger Datencluster. Daher können auch noch mit relativ wenigen Daten Modelle erstellt werden, die in diesen Bereichen akzeptabele Vorhersagen liefern.

1.5 Regressionsmodelle

Es soll ein funktionaler Zusammenhang zwischen der Zielgröße y und den m voneinander unabhängigen Prozessparametern bzw. Eingangsgrößen $x_1, \ldots, x_m$ ermittelt werden. Dazu werden die p freien Anpassungsparameter der Modelle durch Auswertung einer Menge von n gemessenen Datenpunkten

$$D = \left\{ \left(\left(x_1, \ldots, x_m\right)_1, y_1 \right), \ldots, \left(\left(x_1, \ldots, x_m\right)_n, y_n \right) \right\} \tag{1.5.1}$$

derart angepasst, dass die Modellergebnisse $f(x_i)$ die gemessenen Werte y_i an den Stützstellen gut reproduzieren. Weiterhin sollen die in den Messdaten enthaltenen Informationen so weit generalisiert werden, dass Vorhersagen für die Zielgröße auch für die bisher noch nicht durch die Produktion abgedeckten Gebiete des Parameterraums gemacht werden können.

Auf Polynomen basierende Regressionsmodelle finden in vielen verschiedenen wissenschaftlichen Bereichen Anwendungen, etwa in der Medizin [37], Chemie [38], Pharmazie [39], Physik [40], Ökonomie [41], Meteorologie [42], Sozialwissenschaften [43] und weiteren [44]. Die meisten dieser Fälle zeichnen sich aber durch eine niedrigere Dimension und niedrigere Komplexität oder höheres Verhältnis zwischen der Anzahl der zur Verfügung stehenden Daten zur Anzahl der möglichen Größen aus, die das Ergebnis beeinflussen. Vielen dieser Probleme liegen Zusammenhänge zugrunde, die mit linearen Funktionen bereits ausreichend gut approximiert werden können.

1.6 Qualität der Anpassung

Um die Güte der Anpassung an die gemessenen Daten zu bewerten, können etwa die Standardabweichung σ des Fehlers zwischen Modell und Messung oder das dimensionslose Bestimmtheitsmaß

$$R^2 = \frac{\mathrm{Cov}(X,Y)^2}{\mathrm{Var}(X)\mathrm{Var}(Y)} = 1 - \frac{\sum_{i=1}^{n} (y_i - f(x_i))^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \tag{1.6.1}$$

$$\mathrm{Var}(X) := \mathrm{E}\left((X - \mathrm{E}(X))^2\right) = \sigma^2 \tag{1.6.2}$$

$$\mathrm{Cov}(X,Y) := \mathrm{E}\left((X - \mathrm{E}(X)) \cdot (Y - \mathrm{E}(Y))\right) \tag{1.6.3}$$

aus der Kovarianz $\mathrm{Cov}(X,Y)$ und den Einzelvarianzen $Var(X)$, $Var(Y)$ der Eingangsdaten X und Modellwerte Y verwendet werden. E bezeichnet hierbei den Erwartungswert der Verteilung. $0 \leq R^2 \leq 1$ beschreibt, wie viel der Streuung in den Messdaten durch das Modell erklärt wird bzw. wie viel besser das Modell die gemessenen Daten wiedergibt als der Mittelwert $\bar{y}$ aus allen Messdaten. $R^2 = 0$ bedeutet, dass das Modell überhaupt keinen Zusammenhang zu den Messdaten herstellt und $R^2 = 1$, dass es sie exakt abbildet.

Eine gegebene Schar von Datenpunkten kann umso genauer angepasst werden, je mehr freie Parameter p ein Modell zur Verfügung hat. Das ist einerseits nötig, um komplexe Zusammenhänge hinreichend genau modellieren zu können, zum anderen erlaubt es dem Modell die Anpassung an das Rauschen, was zu unvernünftigen Vorhersagen außerhalb der Stützstellen führen kann.

Als Beispiel sei die verrauschte lineare Funktion aus Abschnitt 1.4.6 für eine lineare Anpassung sowie polynomielle Regressionen 3. und 9. Ordnung betrachtet. Die Anzahl der freien Parameter p entspricht dabei der Anzahl unabhängiger Variablen und damit der Ordnung des Regressionspolynoms. Bei steigender Ordnung nimmt die Standardabweichung ab und das Bestimmtheitsmaß zu, bis das Regressionspolynom bei 9. Ordnung genügend Parameter hat, um die zehn Messwerte exakt wiederzugeben. Dabei oszilliert es zwischen den Stützstellen stark, was für hochgradige Polynome typisch und für Inter- und insbesondere Extrapolationen äußerst ungeeignet ist. Das Modell hat die zufälligen Streuungen der Messungen abgebildet und damit die Information über die darunterliegende Funktion $f(x) = x$ unkenntlich gemacht. Dieser Effekt wird als Überanpassung bezeichnet.

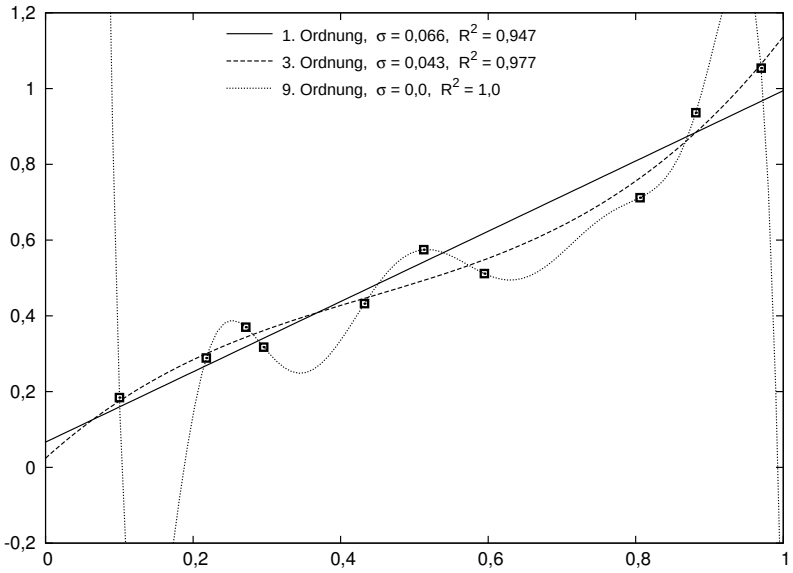


Abbildung 1.3: *Polynomielle Regressionen einer verrauschten linearen Funktion.*

Ein Vergleich der Bestimmtheitsmaße zweier verschiedener Modelle zur Beurteilung ihrer Approximationsqualität kann auch die Anzahl der freien Parameter der Modelle explizit berücksichtigen. Um den Gewinn einer besseren Approximation aufgrund der Erhöhung der freien Modellparameter gegen die dabei wachsende Tendenz der Überanpassung aufzuwiegen, kann das korrigierte Bestimmtheitsmaß [45]

$$R^2_{korr} = 1 - (1 - R^2)\frac{n-1}{n-p-1} \tag{1.6.4}$$

herangezogen werden. Hierbei wird die Anzahl der Messdaten n der Anzahl der Modellparameter p gegenübergestellt, um der durch eine Erhöhung von p erwartungsgemäß besseren Korrelation Rechnung zu tragen. $n-p-1$ sind die Freiheitsgrade der Anpassung, also die Anzahl der Messwerte, die zusätzlich zur mindestens nötigen Anzahl vorhanden sind. Bei einer Erhöhung von p steigt R^2_{korr} nur dann, wenn die Zunahme von R^2 stärker ist als die Abnahme des die erhöhte Komplexität strafenden Terms $(n-1)\,/\,(n-p-1)$.

R^2 sowie R^2_{korr} betrachten lediglich die Abbildung der gemessenen Daten und lassen daher keine direkte Aussage über die Inter- und Extrapolationsfähigkeiten eines Modells zu.

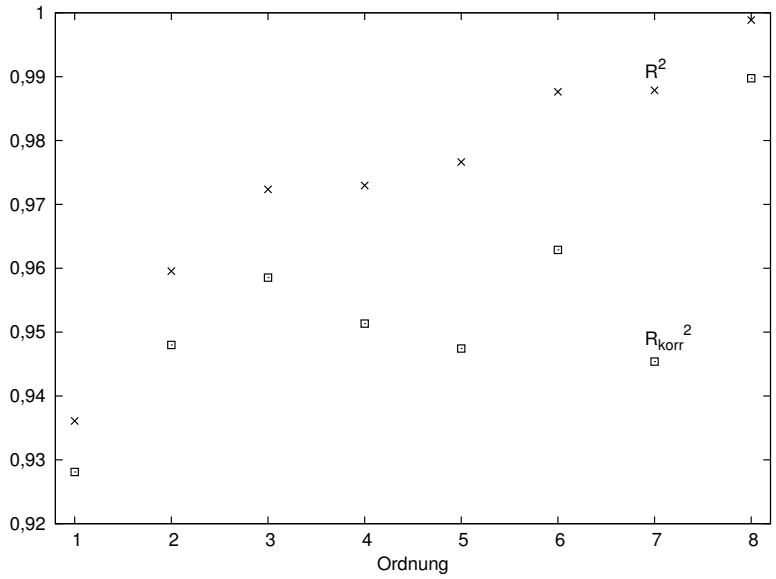


Abbildung 1.4: *R^2 und R^2_{korr} der Funktion aus Abbildung 1.3 als Funktion der Regressionsordnung. Die starke Überanpassung der hochgradigen Polynome ist aus der Betrachtung von R^2_{korr} nicht ersichtlich.*

Abbildung 1.4 zeigt den Verlauf der beiden Bestimmtheitsmaße für verschiedene Regressionsordnungen für die im vorherigen Beispiel genannte Funktion $f(x) = x$ (Abbildung 1.1 auf Seite 11). Hochgradige Anpassungen werden durch R^2_{korr} zwar etwas schlechter bewertet als im unkorrigierten Fall, aber der stark überangepasste Fall 8. Ordnung liefert nach wie vor das höchste R^2_{korr}. Die Form der Anpassungskurve 8. Ordnung variiert aber bei Auslassen jeweils eines Datenpunktes sehr stark, was in einem stabilen Modell nicht passieren darf. Diese Eigenschaft der Überanpassung kann zur Überprüfung herangezogen werden, ob ein Modell auch außerhalb der Stützstellen realistische Ergebnisse liefert.

1.6.1 Kreuzvalidierung

Eine Möglichkeit, die Überanpassung im Rahmen der verfügbaren Daten zu verifizieren, ist die Kreuzvalidierung [46, 47]. Dazu wird das Modell zunächst an eine Teilmenge der Messdaten, der sogenannten Trainingsmenge, angepasst. Anschließend werden die Modellvorhersagen für die restlichen Daten, der Testmenge, mit deren Messwerten verglichen. Dass sich der Abbildungsfehler auf den Testdaten dabei nicht wesentlich gegenüber dem der Trainingsdaten verschlechtert, ist eine notwendige, wenn auch nicht hinreichende Bedingung dafür, dass das Modell nicht überanpasst. Test- und Trainingsdaten sollten dabei statistisch gleichmäßig verteilt sein, um eine gleichmäßige Verteilung der Information zu gewährleisten und eine Verzerrung der Ergebnisse durch Clusterbildung zu vermeiden, die einen unrepräsentativ hohen oder niedrigen Fehler auf einem der Datensätze zur Folge haben könnte. Daher werden die Daten

üblicherweise zufällig auf die einzelnen Datensätze verteilt.

Um die Anpassung an den gesamten Datensatz bewerten zu können, werden die Messdaten $D = \{(x,y)\}$ zunächst in k gleich große disjunkte Teilmengen unterteilt:

$$\begin{aligned} D^{1,\ldots,k} &= \left\{ \left(x_j, y_j\right) \mid j \,\text{Zufallszahl} \in \{1,\ldots,|D|\} \right\}, \\ \left|D^{1,\ldots,k}\right| &= \text{int}\left(\tfrac{|D|}{k}\right), \\ D^i \cap D^j &= \emptyset \;\forall i \neq j \end{aligned} \tag{1.6.5}$$

Das Verfahren wird k-mal wiederholt, wobei jeweils k-1 Teilmengen als Trainingsmenge $D^{(i)}_{train} = D^i$, $i = 1,\ldots,k$ und die verbleibende als Testmenge $D^{(i)}_{test} = D \setminus D^i$ genutzt wird. Zur Bewertung des Modells wird der Mittelwert der in den einzelnen Kreuzvalidierungsschritten erhaltenen mittleren quadratischen Fehler MSE der Modellfunktion $f(x)$ auf die Testdatenmengen $\{D_{test}\}$ herangezogen:

$$MSE_{test} = \frac{1}{k} \sum_{i=1}^{k} \frac{1}{\left|D^{(k)}_{test}\right|} \sum_{j=1}^{\left|D^{(k)}_{test}\right|} \left\| f(x_{i,j}) - y_{i,j} \right\|^2 \tag{1.6.6}$$

Je größer k gewählt wird, desto geringer können sich Unregelmäßigkeiten in den Verteilungen der einzelnen Trainingsmengen auswirken und die Modelle erhalten beim Training immer mehr Information. Mit den kleiner werdenden Testdatensätzen ist dann aber die Voraussetzung, dass gleichmäßig über den gesamten Datensatz verteilte Messungen entnommen wurden, immer schlechter erfüllt. Im Extremfall, der Leave-One-Out-Kreuzvalidierung, besteht die Testmenge nur aus einem einzigen Datenpunkt und die Annahme trifft nicht mehr zu. Die Einzelmodelle tendieren dadurch zum Überanpassen und die abgeschätzten Einzelfehler streuen stark [48]. Die optimale Wahl von k ist deshalb problemabhängig. Ein weiterer Nachteil der Leave-One-Out-Kreuzvalidierung liegt im hohen Rechenaufwand, der allerdings bei linearen Regressionen auf eine einzige Anpassung auf den gesamten Datensatz zurückgeführt werden kann [49].

In der Literatur finden sich Untersuchungen zu pauschal geeignetem k [50], etwa $k = 10$, aber die Erfahrung mit den hier behandelten und ähnlichen Anwendungsfällen hat gezeigt, dass es sinnvoll ist, verschiedene Werte von k zu testen.

Abbildung 1.5 zeigt das Ergebnis einer Leave-One-Out-Kreuzvalidierung für das obige Beispiel. Aufgetragen ist das Verhältnis der über alle Durchläufe gemittelten MSE für Trainings- und Testdaten, MSE_{train}/MSE_{test}. Bei einer starken Überanpassung ist $MSE_{test} \gg MSE_{train}$ und das Verhältnis geht gegen 0. Wenn das Modell gut generalisiert, ist $MSE_{test} \approx MSE_{train}$ und das Verhältnis liegt nahe bei 1. Es sind auch Verteilungen von Test- und Trainingsdaten nicht ausgeschlossen, bei denen die Vorhersagen der Testdaten zu kleineren Fehlern führt als die Anpassung an die Trainingsdaten. In diesem Fall

ist $MSE_{train}/MSE_{test} > 1$. Werte $\gg 1$ sind aber in der Praxis ungewöhnlich und ein Anzeichen für eine ungeeignete Aufteilung der Datensätze.

In diesem Beispiel wird die lineare Regression $(p = 1)$ am besten bewertet, die Ordnungen 2 und 3 mäßig und Ordnung 4 und höher zeigen eine deutlich schlechte Verallgemeinerungsfähigkeit.

Selbst das beste Modell kann nicht mehr Information aus einem Datensatz gewinnen, als durch das Rauschen noch erkennbar ist. Daher ist es nicht möglich, ein absolutes Gütemaß für ein Modell festzulegen, da hierzu sowohl Kenntnis über die Komplexität der zugrundeliegenden Zusammenhänge als auch über die Fehlerverteilung nötig wären. Eine Kreuzvalidierung mithilfe eines Fehlermaßes wie des *MSE* ermöglicht es, den zu erwartenden Testfehler direkt ablesen zu können und somit zu bewerten, ob die Qualität der Anpassung für den gewünschten Zweck ausreichend ist. Eine ausführliche Übersicht der Kreuzvalidierungsstrategien zur Modellselektion findet sich in [51].

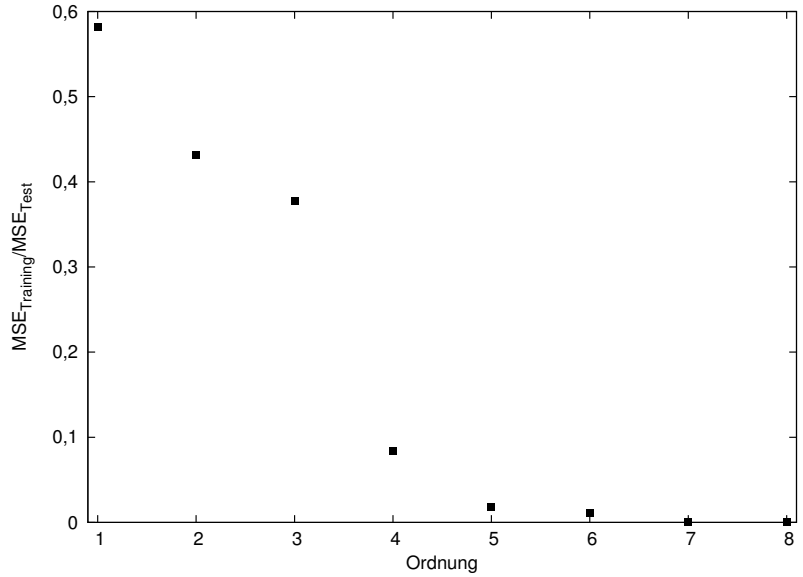


Abbildung 1.5: *Verhältnis des Mean Squared Error der Trainings- und Testdaten für eine Leave-One-Out-Kreuzvalidierung der Regressionen aus Abbildung 1.3.*

1.7 Bestmögliche Anpassungsfähigkeit

Aufgrund von Messfehlern oder Vereinfachungen bei der Prozessbeschreibung (Abschnitt 1.4.1) kann es vorkommen, dass mehrere Messungen mit identischen Eingangsparametern zu unterschiedlichen Messungen führen. Dann existiert keine Funktion, die die Daten exakt ($R^2 = 1$) wiedergeben kann. Der mittlere quadratische Fehler lässt sich zerlegen in

$$
\begin{aligned}
MSE &= \frac{1}{n}\sum_{i=1}^{n}\left(y_i - f(x_i)\right)^2 \qquad (1.7.1)\\
&= \overbrace{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \bar{y}_i\right)^2}^{\text{reiner Fehler}}\\
&+ \overbrace{\frac{1}{n}\sum_{i=1}^{n}|W|\cdot\left(f(x_i) - \bar{y}_i\right)^2}^{\text{mangelnde Anpassung}}\\
\bar{y}_i = \frac{1}{|W|}\sum_{j\in W} y_j \quad &\text{mit} \quad W = \left\{j \mid x_j = x_k \,\forall k\right\}
\end{aligned}
$$

Wegen der Messfehler in den $\{x_i\}$ müssten für eine realistischere Abschätzuung komponentenweise Intervalle angegeben werden, um den Mittelwert der $\bar{y}_i$ zu bilden. Eine Modellfunktion kann also bestenfalls mittlere Anpassungswerte minimieren.

1.7.1 Verzerrung-Varianz-Kompromiss

Wenn ein Modell zu wenig Parameter hat, um die Komplexität der den Daten zugrundeliegenden Funktion zu approximieren, dann wird es einen hohen Verzerrungsfehler (Bias) aufweisen, da die Modellvorhersagen weit von den Zieldaten entfernt liegen. Dies ist z. B. der Fall, wenn eine lineare Funktion $f_1(x)$ zur Approximation von Daten verwendet wird, die von einer gekrümmten Funktion $g(x)$ erzeugt wurden. Der Fehler wird sich aber nicht wesentlich ändern, wenn das Modell zur Vorhersage eines anderen aus $g(x)$ generierten Datensatzes verwendet wird: würde das Polynom wieder an die neuen Daten angepasst, dann würden die Koeffizienten ähnlich ausfallen.

Das Gegenteil wäre ein Modell $f_2(x)$ mit zu vielen Parametern, bei dem die Überanpassung zu einem geringen Anpassungsfehler führt. Werden neue Daten aus $g(x)$ generiert, dann wird der Vorhersagefehler von $f_2(x)$ groß. Das Modell hat dann einen großen Varianzfehler. Würde das gleiche Polynom wieder an die Daten angepasst, dann würden sich deutlich unterschiedliche Koeffizienten ergeben.

Unter Shrinkage wird der Effekt verstanden, dass das Bestimmtheitsmaß bei der Vorhersage von Daten, die nicht zum Trainieren eines Modells verwendet wurden, üblicherweise gegenüber dem Bestimmtheitsmaß der Anpassung an die Trainingsdaten abnimmt. Dieser Effekt ist umso mehr ausgeprägt, je stärker das Modell überanpasst. R^2_{korr} (Gleichung 1.6.4) ist ein Schätzer für den Shrinkage-Effekt. Die Suche nach einer guten Anpassungsfunktion lässt sich als Problem der Minimierung der Summe von Varianz- und Verzerrungsfehler ausdrücken (Bias-Variance-Tradeoff).

1.8 Multikollinearität

Multikollinearität liegt vor, wenn verschiedene Eingangsparameter x_i und x_j fast linear abhängig sind. Sie enthalten dann die gleiche Information und bringen bei der Modellentwicklung keinen Erkenntnisgewinn. Statt dessen können sie zu numerischen Problemen bei der Anpassung von Modellen führen [52]. Linear korrelierte Eingangsparameter können durch Auftragung gegeneinander oder Berechnung ihres R^2 ermittelt werden. Multikollinearität liegt für $R^2(x_i, x_j) \approx 1$ vor.

Stark korrelierte Eingangsparameter sind bezüglich ihres Einflusses auf das Ergebnis statistisch nicht unterscheidbar und sollten daher nicht gleichzeitig in die Modellbildung einbezogen werden. Statt dessen sollten korrelierte Variablen x_i und x_j im Voraus identifiziert werden. Je eine der paarweise korrelierten Variablen, z. B. x_i, kann dann entfernt werden oder es kann eine kombinierte Variable aus beiden gebildet werden. Bei der Interpretation der Modellergebnisse muss dann aber beachtet werden, dass der statistisch ermittelte Einfluss von x_j tatsächlich auch auf x_i oder einer Kombination von x_i und x_j zurückzuführen sein könnte. In diesem Fall kann für eine realistische Interpretation der Modellwerte nicht auf weiteres Expertenwissen der wahren zugrundeliegenden Zusammenhänge verzichtet werden.

Fehlerfortpflanzug während der Modellberechnung kann eine weitere Ursache für Kollinearitätseffekte sein [53] und kann insbesondere bei numerisch nicht stabilen Rechenmethoden zu Problemen führen.

1.9 Heteroskedastizität

Heteroskedastizität liegt vor, wenn die Streuung der Fehler der Messdaten von den Eingangsparametern x_i abhängt. Ursachen hierfür können unzureichend genaue Modelle oder systematische Fehler in den Messungen sein. Durch systematische Fehler, die zu unsymmetrischen Fehlerverteilungen führen, kann die Modellbildung verzerrt werden. Durch die Auftragung der Residuen r_i gegen die x_i kann auf Heteroskedastizität geprüft werden. Wenn die Residuen gleichmäßig verteilt sind, spricht man von Homoskedastizität.

1.10 Bewertung der Signifikanz der Prozessparameter

Die Wahl der Eingangsparameter sollte immer in zwei Stufen erfolgen. Zuerst sollten mithilfe von Expertenwissen aus allen verfügbaren Parametern diejenigen ausgewählt werden, von denen eine Wirkung auf die Zielgröße bekannt ist oder vermutet wird. Falls möglich sollte die Anzahl der Parameter dann noch durch Kombinationen reduziert werden, beispielsweise mittels einer Dimensionsanalyse oder durch Erstellung von Kennzahlen aus physikalischen Modellen.

Danach kann die Signifikanz der als relevant vermuteten Prozessparameter $x_1, \ldots, x_m$ zur Beschreibung der Zielgröße y im Rahmen eines Modells überprüft werden, indem die gesamte Modellentwicklung m-mal sukzessive unter Auslassung jeweils eines Prozessparameters x_i wiederholt wird. Das Bestimmtheitsmaß R_i^2 der Anpassung ohne x_i ist dann ein Maß für dessen Signifikanz: je kleiner R_i^2, also je schlechter die Anpassung $f(x_1, ..., x_{i-1}, x_{i+1}, ..., x_m)$ ausfällt, desto besser ist x_i zur Anpassung bezüglich der gewählten Anpassungsordnung geeignet. Da bei den m sukzessiven Testdurchläufen sowohl die Eingangsdatenmenge als auch die Anzahl der Eingangsvariablen gleich ist, kann als Signifikanzmaß ebenso die Standardabweichung oder der mittlere quadratische Fehler gewählt werden. Durch diese Vorauswahl kann ein großer Satz von Prozessparametern auf einen kleineren repräsentativen Satz reduziert werden, indem diejenigen Parameter verworfen werden, welche die geringste Verbesserung zum Modell beitragen.

Kapitel 2

Datensätze

2.1 Empirische Daten

Zur Anwendung der Modelle wurden drei für die Stahlproduktion interessante Datensätze ausgewählt. Diese weisen verschiedene Charakteristika auf, die besondere Anforderungen an die Modelle stellen. In Abschnitt 2.1.1 wird ein Datensatz vorgestellt, der aus verhältnismäßig vielen gut reproduzierbaren Messungen besteht. Abschnitt 2.1.2 befasst sich mit einem Datensatz, bei dem die Messungen stark streuen und durch einen Schwellwert beschränkt sind. Abschnitt 2.1.3 beschreibt einen Datensatz, bei dem nur wenige Messungen vorhanden sind und der Satz an Eingangsgrößen möglicherweise unvollständig ist oder das System chaotisches Verhalten zeigt.

2.1.1 Zugfestigkeit

Beim Zugversuch [10] werden Festigkeits- und Verformungskennwerte einer Stahlprobe bestimmt. Dazu wird eine Probe unter Einwirkung einer kontinuierlich ansteigenden Kraft bis hin zum Bruch gedehnt. Während der Dehnung wird die relative Längenänderung gemessen und gegen die Spannung bezüglich der Querschnittsfläche der undeformierten Probe aufgetragen (Spannungs-Dehnungs-Diagramm).

Bei kleinen Spannungen verhält sich die Stahlprobe linear elastisch. Bei Erreichen der Streckgrenze R_e beginnt die Probe, sich plastisch zu verformen. Dieser Teil der Verformung ist irreversibel. Der genaue Verlauf der Spannungskurve im plastischen Bereich ist werkstoffabhängig. Da sich der Probenquerschnitt während der weiteren Dehnung einschnürt, die Spannung aber bezüglich des Ausgangsquerschnitts berechnet wird, nimmt die aufgetragene Spannung mit weiterer Zugkraft bis zum Bruch der Probe hin ab. Die maximal auftretende Spannung wird als Zugfestigkeit R_m bezeichnet. In der Praxis soll das Modell Vorhersagen machen, so dass Vorgaben $R_{e,min} \leq R_e \leq R_{e,max}$ und $R_{m,min} \leq R_m \leq R_{m,max}$ und ggf. weitere

Bedingungen wie $R_e/R_m \leq Limit$ eingehalten werden.

Der Datensatz der Zugfestigkeitswerte enthält 26.203 Messungen über einen weiten Bereich verschiedener Stahlgüten. Als Eingangsparameter wurden die chemische Zusammensetzung (14 Parameter), die zur Reinheitsbehandlung zugegebene CaSi-Menge, der Verformungsgrad und eine Temperatur bei der letzten Walzung, eine Temperatur am Ende der Kühlung und die Blechdicke genommen, insgesamt 19 Parameter. Die Einzelmessungen der Zugfestigkeitswerte streuen relativ gering, so dass hier auch Modelle gute Resultate liefern können, die sensibel gegenüber verrauschten Datensätzen sind.

In der Produktion werden je nach Anforderungen unterschiedliche Prozessabläufe beim Walzen und Kühlen angewendet, die dazu führen können, dass gegebene Eingangsparameter zu zwei verschiedenen Prozessen nicht direkt vergleichbar sind. Das Walzen erfolgt beispielsweise bei einem Prozesstyp oberhalb der Austenit-Ferrit Umwandlungstemperatur, bei einem anderen unterhalb dieser Temperatur. Der Eingangsparameter „Umformgrad" hat daher in beiden Fällen eine unterschiedliche Auswirkung auf die weitere Gefügeentstehung. Die Kühlung kann in mehreren Schritten mit oder ohne Pausen erfolgen, dabei kann durch Variation der Wassermenge sowohl schnell als auch langsam gekühlt werden. Zur genauen Modellierung dieser Abläufe müsste eine Vielzahl weiterer Parameter eingeführt werden, die jeden einzelnen Zwischenschritt charakterisieren. Dadurch würde die Komplexität der Daten stark erhöht. Da aber die Abläufe innerhalb der einzelnen Prozessklassen immer ähnlich sind, ist es naheliegend, statt dessen separate Modelle für jede Prozessklasse zu erstellen und somit mit weniger Eingangsdaten auszukommen und die Vergleichbarkeit der einzelnen Prozessparameter zu gewährleisten.

Die Wahl passender Eingangsparameter erfolgte empirisch aus der Menge aller in der Produktion aufgenommenen Prozessparameter. Dazu wurde zunächst ein Satz von Parametern basierend auf Expertenwissen ausgewählt, woraus dann mehrere Modelle nacheinander jeweils unter Vernachlässigung eines Parameters erstellt und bewertet wurden. Bei deutlicher Verschlechterung der Modellgüte wurde der vernachlässigte Parameter als für die Modellierung signifikant betrachtet und beibehalten.

In der Literatur existieren Untersuchungen zur Abhängigkeit der Festigkeit vom Gehalt einzelner Legierungselemente [10]. Weiterhin wurde die Wirkung von Umformung und Kühlung auf die Gefügebildung und deren Einfluss auf die Festigkeit untersucht. Dadurch können in Spezialfällen qualitative Vergleiche zwischen Modellvorhersagen und Literaturmessungen gemacht werden. Es kann beispielsweise die Änderung der Zugfestigkeit bei Variation eines einzelnen Legierungselements untersucht werden, während die restlichen Eingangsgrößen konstant vorgegeben werden. Die direkte quantitaive Übertragung von Modellvorhersagen auf Literaturmessungen ist aber nicht möglich, da die Laborversuche nicht den gesamten Produktionsprozess abbilden und meistens Stähle mit weniger Legierungselementen beschreiben als in der Produktion üblich. Die hohe Dimensionalität der Probleme macht systematische Versuchsreihen über den gesamten Parameterraum unmöglich.

2.1.2 Wasserstoffinduzierte Rissbildung (HIC)

Der Prozess der Wasserstoffversprödung (Hydrogen Induced Cracking, HIC) von Stahl spielt bei Druckbehältern und sauergasführenden Pipelines eine wichtige Rolle. Dabei diffundiert Wasserstoff in den Stahl und kann zu Versprödungsrissen führen [54], die ein Materialversagen zur Folge haben können. So kamen beispielsweise im Juli 1984 17 Menschen bei einer Explosion eines Druckbehälters in Chicago ums Leben, bei der die wasserstoffinduzierte Rissbildung eine Rolle spielte [55].

Um den Versprödungsprozess zu simulieren und charakterisieren, wird eine Stahlprobe über eine bestimmte Zeit in eine saure Lösung getaucht von Wasserstoff umspült. Je nach Widerstandsfähigkeit der Stahlprobe gegenüber HIC entstehen dabei mehr oder weniger große und viele Versprödungsrisse. Diese werden nach der Korrosionsbehandlung vermessen. Dabei wird unter anderem die relative Rissfläche (Crack Area Ratio, *CAR*) bestimmt. Je größer der *CAR*-Wert, desto schlechter wird die Probe bezüglich HIC-Anfälligkeit eingestuft. Bleche, die für Produkte verwendet werden, die in Sauergasumgebungen Anwendung finden, müssen HIC-resistent sein. Um dies sicherzustellen, dürfen die *CAR*-Werte und ggf. weitere HIC-Charakteristika von Proben dieser Bleche ein vorgegebenes Limit nicht überschreiten. Abbildung 2.1 zeigt eine Aufnahme von HIC-Rissen in einer Probe.

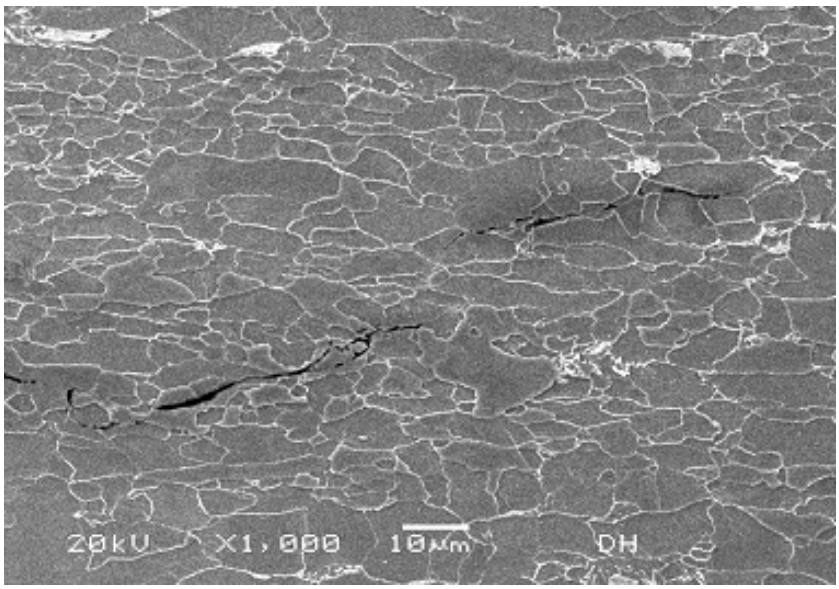


Abbildung 2.1: *Rasterelektronenmikroskopische Aufnahme von HIC-Rissen aus [56]. Die Probe wurde nach der HIC-Behandlung poliert und geätzt, um die Strukturen deutlicher darstellen zu können. Die HIC-Risse erscheinen auf der Aufnahme schwarz, die Grenzen der Kristallite weiß.*

In Abbildung 2.2 ist ein Beispiel einer Auftragung von *CAR*-Werten gegen einen Prozessparameter dargestellt. Die Streuung der Messdaten ergibt sich zum einen aus Messfehlern und zum anderen aus

der Variation der restlichen Prozessparameter, da hier nicht identische, sondern nur ähnliche Produkte verglichen wurden.

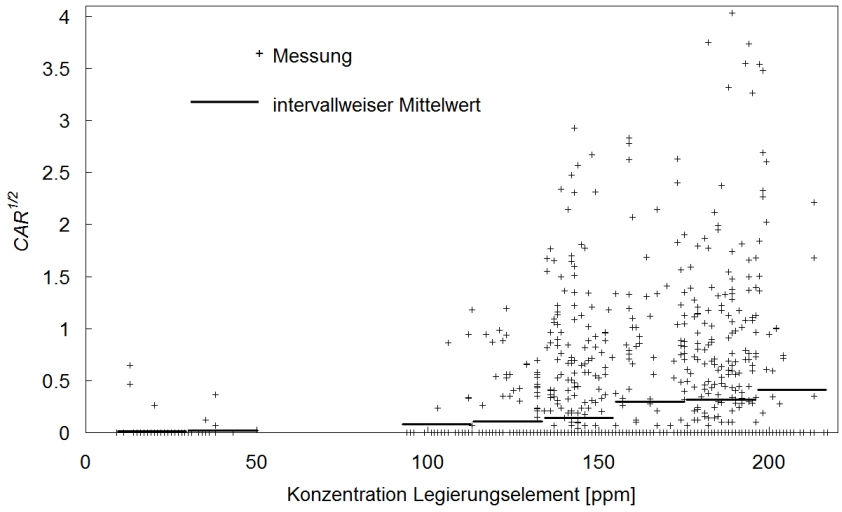


Abbildung 2.2: *Messungen des CAR in Abhängigkeit eines Legierungselements für einen speziellen Walz- und Kühlprozess. Um die Streuung zu verringern, wurde ein intervallweiser Mittelwert abgebildet. Dieser zeigt eine Zunahme des CAR-Werts für Legierungselementkonzentrationen > 100 ppm. Die gemessenen Werte im Konzentrationsbereich < 100 ppm sind bis auf ein Restrauschen der Messung Null.*

Bei guten Proben entstehen keine Risse, d. h. $CAR = 0$. In diesem Fall lässt sich aber nicht entscheiden, ob bereits eine geringfügige Variation der Prozessparameter zu $CAR > 0$ geführt hätte oder ob die Probe selbst bei größerer Variation immer noch keine Risse gezeigt hätte. Dementsprechend können zwei unterschiedliche Proben, die beide $CAR = 0$ ergeben haben, durchaus als unterschiedlich HIC-resistent betrachtet werden.

Es kann zwar versucht werden, durch eine Verschärfung der Testbedingungen mehr HIC-Risse zu erzwingen [56] und somit mehr Information aus Proben zu erlangen, die ursprünglich $CAR = 0$ ergaben. Dabei sind aber auch prozesstechnische Grenzen gegeben wie minimaler pH-Wert der Lösung oder maximale Exponierungsdauer der Probe in der Lösung. Durch eine Verschärfung kann auch nicht garantiert werden, dass alle betrachteten Stahlgüten früher oder später Risse aufweisen. Weiterhin bestünde dann die Gefahr, dass schlechte Proben bei zu langer Exponierungsdauer eine Sättigung bei hohen *CAR*-Werten erfahren, wodurch Informationen über HIC-ungeeignete Prozessbedingungen verloren gingen. Letztendlich müsste überprüft werden, ob die Verschärfung der Testbedingungen auch für die realen Prozesse repräsentativ ist.

Diese intrinsische Beschränkung des HIC-Informationsgehalts durch die Messung des *CAR* stellt eine Einschränkung bei der Weiterentwicklung von Stählen dar, da hier nach Prozessparametern für Stahlqualitäten gesucht wird, die unter den unvermeidbaren Prozessstreuungen immer noch robuste Ergebnisse

liefern. Weiterhin soll genügend Raum für Variationen einzelner Prozessparameter vorhanden sein, um den Anforderungen an weitere Materialeigenschaften gerecht werden zu können.

Die relative Fläche *CAR* der HIC-Risse innerhalb der Proben wird als zu modellierende Zielgröße betrachtet. Sie ist durch die Bedingung $CAR \geq 0$ abgeschnitten, was im Folgenden als „Capped Data"-Eigenschaft bezeichnet wird.

Die 19 Eingangsparameter wurden ähnlich den Zugfestigkeitsdaten für ein bestimmtes Walz- und Kühlverfahren aus chemischer Zusammensetzung, Temperaturen und Umformparametern und für eine festgelegte Probenlage im Blech und Probenbehandlung zur Simulation des Korrosionsprozesses gewählt. Der Datensatz besteht aus 1578 *CAR*-Messwerten, ist also für die hohe Dimensionalität nur dünn besetzt. Der Wertebereich der *CAR*-Messungen liegt zwischen 0% und 20%, wobei ein Großteil der Messwerte allerdings bei $CAR < 1\%$ liegt. Damit diese kleinen Werte bei Optimierung des *MSE* nicht unverhältnismäßig schwach gewichtet werden, wurde bei der Anpassung die transformierte Größe $y' = \sqrt{CAR}$ verwendet.

Der HIC-Datensatz zeichnet sich durch stark verrauschte Daten aus. Daher eignet er sich als Härtetest für Modelle, die aus wenigen ungenauen Daten möglichst viel Information entnehmen sollen, ohne überanzupassen. Das Entstehen, Ausbreiten und Stoppen von HIC-Rissen hängt nicht nur von den globalen Eigenschaften der Stahlmatrix ab, d. h. der chemischen Zusammensetzung sowie der Beschaffenheit und Größe der Kristallite, sondern auch von lokalen Inhomogenitäten wie Einschlüssen oder Ausscheidungen verschiedener Partikel. Somit wäre für eine repräsentative Bestimmung des *CAR*-Wertes eine hohe Anzahl Proben nötig, was bei zerstörenden Prüfverfahren aber unwirtschaftlich ist. Mehrere Proben an Blechen zu sonst gleichen Prozessparametern können daher stark streuende *CAR*-Werte aufweisen und es kann insbesondere bei Entnahme von wenigen Proben vorkommen, dass Bleche mit schlechten HIC-Eigenschaften durch zufälliges Treffen von Stellen mit $CAR = 0$ unrepräsentativ gut bewertet werden, während gute Bleche durch Proben an schlechten Stellen zu schlecht bewertet werden.

2.1.3 Oszillationsmarken

Beim Stranggießen von Stahl [15] wird der flüssige Stahl in eine Kokille gefüllt, an deren Berandung er erstarrt und nach unten herausgeführt wird. Dadurch wird ein kontinuierlich austretender Stahlstrang erzeugt, der unter weiterer Kühlung allmählich von außen nach innen erstarrt. Damit der Stahl beim Erstarren nicht an der Kokille haftet, oszilliert diese. Oszillationsfrequenz und Oszillationsamplitude sowie die Form der Schwingung, üblicherweise sinusförmig, müssen an die Gießgeschwindigkeit des Stahls bei vorgegebener Kokillendimensionierung angepasst werden, um eine hinreichend feste erstarrte Schale zu gewährleisten. Dabei spielen die chemische Zusammensetzung des Stahls und die Gießtemperatur ebenfalls eine Rolle.

Die oszillierende Kokille führt bei der Erstarrung zu einer Verformung der Strangschalenspitze, die sich durch Rillen auf der Strangoberfläche bemerkbar machen kann. Wenn sie zu tief sind, können diese Oszillationsmarken im weiteren Verlauf der Produktion problematisch sein und sollen daher vermieden werden.

Bei Werksversuchen wurde ein Verhalten der Tiefe und des Abstands der Oszillationsmarken beobachtet, das sich nicht mit den Erkenntnissen der Literatur deckt, wenn man von den in der Literatur üblicherweise höheren Gießgeschwindigkeiten auf die bei der Dillinger Hütte verwendeten niedrigen Gießgeschwindigkeiten extrapoliert [21]. Nach den bisherigen Theorien zur Bildung von Oszillationsmarken sollten sich deren Eigenschaften in einfacher Weise aus einigen wenigen Gießparametern ableiten lassen. Es zeigte sich aber, dass sich unter den bei der Dillinger Hütte vorliegenden Gießbedingungen ein komplexes Verhalten einstellt, das sich nicht mit den bisherigen Erwartungen deckt [22]. Dies wurde zum Anlass genommen, die Versuchsdaten empirisch und statistisch zu modellieren.

Der Versuchsdatensatz besteht aus 695 Messungen der Tiefe der Oszillationsmarken. Hierbei wurden Daten zu gleicher Oszillationsamplitude gewählt und vier Eingangsparameter als signifikant angenommen: Oszillationsfrequenz, Gießgeschwindigkeit, Gießtemperatur und nach derzeitigem Wissen stellvertretend für die in diesem Fall relevanten chemischen Eigenschaften des Stahls das Kohlenstoffäquivalent [10], das sich durch eine Linearkombination einiger Legierungselemente ergibt.

2.2 Synthetische Benchmark-Funktionen

Die Generalisierungsfähigkeit eines Modells kann validiert werden, indem es nach Abschluss des Lernens Vorhersagen für Messpunkte macht, die ihm während des Lernens nicht präsentiert wurden. Wenn die vorherzusagenden Punkte komponentenweise in den gleichen Intervallen liegen wie die Lerndaten, dann wird die Generalisierungsfähigkeit bezüglich der Interpolation bewertet, andernfalls bezüglich der Extrapolation. Das setzt voraus, dass die Daten in hinreichend mächtige Trainings- und Validierungsdatensätze aufgeteilt werden können. Der Trainingsdatensatz muss dabei genügend Information zur Erstellung eines aussagekräftigen Modells enthalten. Im Validierungsdatensatz darf das zu bewertende Gebiet nicht zu dünn mit Messpunkten belegt sein, da ein eventuell vorhandenes Rauschen sonst zu unverlässlichen Bewertungen führen kann.

Eine derartige Aufteilung der in dieser Arbeit betrachteten empirischen Datensätze ist nur begrenzt möglich, da sie nicht umfangreich genug sind, um genügend Validierungsdaten bei gleichzeitig ausreichend großer Lerndatenmenge bereitzustellen. Es ist auch keine geschlossene analytische Form zur Berechnung solcher Größen bekannt, die alle Eingangsparameter explizit berücksichtigt und somit zur Generierung neuer Punkte verwendet werden könnte.

Um die Anpassungsqualität in praxisähnlichen Fällen auch außerhalb der Menge der Messdaten bewerten zu können, wurden künstliche Datenmengen mithilfe synthetischer Funktionen generiert. Diese Benchmark-Funktionen stellen keine Simulation der physikalischen und chemischen Grundlagen während des Herstellungsprozesses dar, sondern beschreiben nur einige der aus den realen Datensätzen bekannten funktionalen Eigenschaften qualitativ. Dadurch können beliebig große Mengen an Datenpunkten generiert werden, mit denen verschiedene Modelle trainiert und überprüft werden können.

Die synthetischen Funktionen enthalten polynomielle Terme der Eingangsparameter bis zur Ordnung 3 und Produkte sowie Verhältnisse daraus, um lineare Abhängigkeiten, Krümmungen, Wendepunkte und Wechselwirkungen zu simulieren. Durch sigmoidale Anteile sollen Übergangsphänomene dargestellt werden und die Werte einiger Funktionen wurden durch Schwellwerte beschnitten (Capped Data) . Damit wurden Datensätze mit $m = 2$, $m = 10$ und $m = 19$ Dimensionen erstellt. Trainings- und Validierungsdatensätze bestanden bei den 2- und 10-dimensionalen Funktionen aus $n = 500$ Datenpunkten. Bei der 19-dimensionalen Funktion enthielten sie jeweils 1578 Datenpunkte, um einen Vergleich mit dem HIC-Problem zu ermöglichen.

Um die Datenverteilung einer realen Verteilung anzunähern, wurden sowohl gleichverteilte als auch in Clustern angeordnete Eingangsdaten betrachtet. Die Daten einiger Funktionen wurden durch zufällige Streuungen verrauscht.

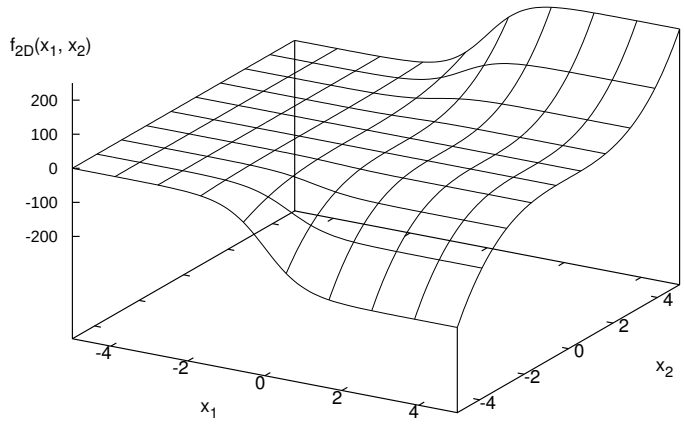


Abbildung 2.3: *2-dimensionale Benchmark-Funktion 1 mit* tanh- *und kubischem Anteil.*

Benchmark-Funktion 1 (2D), Abbildung 2.3:

$$f_1(x_i) = (\tanh(x_1) + 1) \cdot x_2^3, \quad -5 \leq x_1, x_2 \leq 5 \tag{2.2.1}$$

Benchmark-Funktion 2 (10D):

$$f_2(x_i) = (\tanh(x_1) + 1) \cdot x_2^3 + x_3 \cdot x_4 \cdot x_5^2 + \frac{1}{1+\exp(-x_6)} \cdot \frac{1}{1+\exp(-x_7)} \cdot \frac{x_8 \cdot x_9}{x_{10}+6}, \quad -5 \leq x_i \leq 5 \tag{2.2.2}$$

Benchmark-Funktion 3 (10D, verrauscht):

$$f_3(x_i) = \varepsilon_0 + (1 + \varepsilon_1) \cdot f_2(x_i), \quad -5 \leq x_i \leq 5,\ \varepsilon_i = \text{gleichverteilte Zufallszahl } \in [-0,1; 0,1] \tag{2.2.3}$$

Benchmark-Funktion 4 (10D, Capped):

$$f_4(x_i) = \max\{0; f_3(x_i)\}, \quad -5 \leq x_i \leq 5 \tag{2.2.4}$$

Benchmark-Funktion 5 (10D, Capped, verrauscht):

$$f_5(x_i) = \max\{0; \varepsilon_0 + (1 + \varepsilon_1) \cdot f_3(x_i)\}, \quad -5 \leq x_i \leq 5 \text{ aus Cluster},\ \varepsilon_i = \text{gleichverteilte Zufallszahl } \in [-0,1; 0,1] \tag{2.2.5}$$

Benchmark-Funktion 6 (10D, Capped, verrauscht, in Clustern angeordnet):

$$f_6(x_i) = \max\{0; \varepsilon_0 + (1 + \varepsilon_1) \cdot f_3(x_i)\}, \quad -5 \leq x_i \leq 5 \text{ aus Cluster},\ \varepsilon_i = \text{gleichverteilte Zufallszahl } \in [-0,1; 0,1] \tag{2.2.6}$$

Benchmark-Funktion 7 (19D, Capped, verrauscht, in Clustern angeordnet):

$$
\begin{aligned}
f_7(x_i) &= \max\{0;\, \varepsilon_0 + (1+\varepsilon_1)\cdot f_{19D}(x_i)\}, \\
f_{19D}(x_i) &= (\tanh(x_1)+1)\cdot x_2^3 + x_3\cdot x_4\cdot x_5^2 + \tfrac{1}{1+\exp(-x_6)}\cdot\tfrac{1}{1+\exp(-x_7)}\cdot\tfrac{x_8\cdot x_9}{x_{10}+6} \\
&\quad + \tfrac{x_{11}+6}{x_{12}+6} - x_{13}\cdot x_{14} + \tfrac{x_{15}}{\sqrt{x_{16}+6}} + (x_{17}\cdot x_{18})\cdot\exp(-x_{19}-6), \\
-5 \le x_i &\le 5 \text{ aus Cluster}, \ \varepsilon_i = \text{gleichverteilte Zufallszahl } \in [-0,1;\, 0,1]
\end{aligned}
\tag{2.2.7}
$$

Die Normalverteilung der x_i erfolgte hierbei mit einer Streuung $\sigma = 2$, also gemäß der Wahrscheinlichkeitsdichte $\omega(x_i) = \frac{1}{2\sqrt{2\pi}}\cdot\exp\left(-\frac{1}{2}\left(\frac{x}{2}\right)^2\right)$.

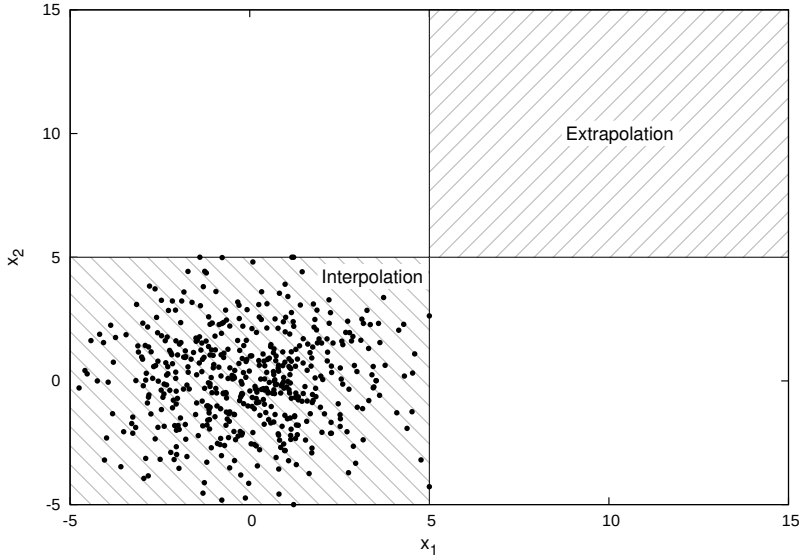


Abbildung 2.4: *Clusterung der Trainingsdaten für x_1 und x_2 zur Berechnung der Benchmark-Funktion 6. Die Bereiche zur Generierung der gleichverteilten Verifizierungsdaten zur Bewertung der Interpolation und Extrapolation sind schraffiert dargestellt.*

Zum Training der Modelle wurde ein Trainingsdatensatz verwendet. Die abschließende Bewertung des Modells erfolgte auf zwei dazu disjunkten Validierungsdatensätzen mit gleichverteilten Daten: ein Satz zur Bewertung der Interpolationsfähigkeit im Trainingsintervall $[-5;\, 5]$ und ein Satz zur Bewertung der Extrapolationsfähigkeit im Intervall $[5;\, 15]$.

Kapitel 3

Adaptive Polynommodelle

Die Grunlagen der multiplen Polynomregression sind im Appendix (Abschnitt 7.1) erläutert. Im Folgenden werden Verfahren betrachtet, die Polynommodelle ermitteln, deren Termstruktur dem Problem angepasst ist. Die Dimension des Problems lässt es nicht zu, alle möglichen linearen Regressionen mit vertretbarem Aufwand zu berechnen und die beste auszuwählen. Statt dessen kommen Algorithmen in Frage, die relativ wenige Regressionsmodelle gezielt konstruieren und das beste davon auswählen.

3.1 Bisherige Ansätze

3.1.1 Signifikanztests

Signifikanztests entscheiden, ob sich statistisch ermittelte Merkmale wesentlich voneinander unterscheiden oder ob der Unterschied nur zufälliger Natur ist. Ein solches statistisches Merkmal ist beispielsweise die Streuung der Residuenverteilung eines Modells (Beispiel Abbildung 3.1 a)). Je kleiner die Streuung ausfällt, desto genauer gibt das Modell die Daten wieder. Aufgrund zufälliger Schwankungen der Stichproben werden sich die Residuenstreuungen zweier verschiedener Modelle aber fast immer voneinander unterscheiden. Daher stellt sich die Frage, ab welcher Differenz der Residuenstreuungen ein signifikanter Unterschied der Modellgüte angenommen werden kann, oder analog, mit welcher Wahrscheinlichkeit eine gegebene Differenz signifikant ist.

Die Residuen einer Modellfunktion $f(x)$ über einen Datensatz der Mächtigkeit n,

$$\begin{gathered}\{(x_i, y_i) \mid i = 1, ..., n\} \\ \{r_i = f(x_i) - y_i \mid i = 1, ..., n\}\end{gathered} \tag{3.1.1}$$

sollen so gewählt werden, dass sie einer zufälligen Stichprobe entsprechen. Dies wäre z. B. nicht der Fall,

wenn überproportional viele Daten aus einem Cluster entnommen wurden. Selbst bei einer zufälligen Auswahl der Daten kann dies nie ganz ausgeschlossen werden.

Falls die Residuen homoskedastisch sind, nähert sich ihre Verteilung mit wachsender Anzahl Messungen einer Normalverteilung an. Da die Normalverteilung nie Null wird, überlappen zwei unterschiedliche Verteilungen immer, so dass selbst bei zwei stark unterschiedlichen Mittelwerten und geringen Streuungen eine endliche, wenn auch sehr kleine, Wahrscheinlichkeit besteht, dass zwei zufällig aus ihnen gewählte Werte sich nicht unterscheiden, obwohl die zugrundeliegenden Verteilungen unterschiedlich sind. Ebenso kann es sein, dass sie sich unterscheiden, obwohl die Verteilungen gleich sind. Dies gilt auch für aus diesen Verteilungen ermittelte Werte, etwa den *MSE*, der aus der Anpassung an verschiedene Datensätze ermittelt wurde.

Um zu bestimmen, ob sich die Residuen zweier Modelle statistisch signifikant unterscheiden, wird ein Signifikanzniveau α vorgegeben. Dieses legt die Wahrscheinlichkeit für die Fehlentscheidung fest, die Residuenverteilungen der Modelle als unterschiedlich zu betrachteten, obwohl sie in Wirklichkeit gleich sind. Ein gängiger Wert hierfür ist $\alpha = 5\%$ [57]. Das bedeutet, dass bei einer stichprobenartigen Ermittlung der Residuen in 95% der Fälle korrekt erkannt wird, dass die Residuenverteilungen der beiden Modelle gleich sind und in 5% der Fälle fälschlicherweise angenommen wird, dass sie unterschiedlich seien. Diese Betrachtungen gelten unter der Annahme, dass die $\{r_i\}$ in hinreichend guter Näherung normalverteilt sind. Bei kleinen Stichprobengrößen wird diese Abschätzung ungenau. Bei nicht normalverteilten Residuenverteilungen, etwa beim HIC-Problem mit Cap-Eigenschaft (Abschnitt 2.1.2), kann sie nicht gemacht werden (Abbildung 3.1).

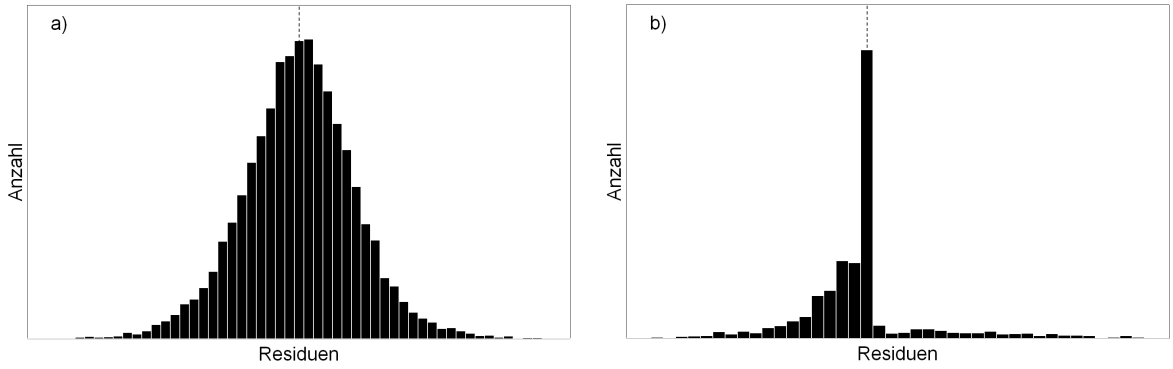


Abbildung 3.1: a) Annähernd normalverteilte Residuen für ein Zugfestigkeits-Modell b) Aufgrund der Cap-Eigenschaft nicht normalverteilte Residuen für ein HIC-Modell. Die gestrichelte Linie markiert die Null.

Die Wahl eines angemessenen Signifikanzniveaus ist problemabhängig. Durch ungeeignete Wahl eines pauschalen Niveaus können fehlerhafte Vergleiche gezogen werden [58].

Da eine höhere Anzahl an Modellparametern auch immer zu einer geringeren Residuenstreuung auf den Trainingsdaten führt, soll untersucht werden, ob die Verbesserung der Abbildungsgüte hinreichend groß ist gegenüber der Erhöhung der Modellkomplexität.

3.1.1.1 F-Test

Beim F-Test [59] wird überprüft, ob sich zwei Normalverteilungen bezüglich eines vorgegebenen Signifikanzniveaus α in ihrer Varianz unterscheiden. Damit kann ermittelt werden, ob ein Modell eine signifikant geringere Varianz der Fehlerverteilung aufweist als ein Vergleichsmodell und folglich die Daten tatsächlich besser wiedergibt und nicht aufgrund zufälliger Schwankungen. Der F-Wert beschreibt das Verhältnis der erklärten zur unerklärten Varianz. Die erklärte Varianz beschreibt die Varianz zwischen den beiden Modellen und die unerklärte Varianz beschreibt die Varianz innerhalb der Modelle:

$$F = \frac{\frac{RSS_1 - RSS_2}{p_2 - p_1}}{\frac{RSS_2}{n - p_2}} \tag{3.1.2}$$

mit den Residuenquadratsummen RSS der Modelle M_1 und M_2

$$RSS = \sum_{i=1}^{n} r_i^2 \tag{3.1.3}$$

Unter der Annahme, dass keins der Modelle die Daten besser erklärt als das andere, folgt der F-Wert einer F-Verteilung mit $(p_2 - p_1, n - p_2)$ Freiheitsgraden. Ist der berechnete F-Wert größer als der zu einem gegebenen α gehörige Wert der F-Verteilung, wird davon ausgegangen, dass ein signifikanter Unterschied zwischen den Modellen besteht.

Streng genommen muss für diese Annahme das eine Modell in dem anderen enthalten sein. Sei $p_2 > p_1$ und alle Terme von M_1 seien auch, mit unterschiedlichen Regressionskoeffizienten a_i, in M_2 enthalten. M_2 geht also durch Hinzufügen von Termen aus M_1 hervor. Damit steigt die Anpassungsflexibilität von M_2 und es ist in der Lage, die Lerndaten mindestens so gut wiederzugeben wie M_1. Mit dem F-Test wird abgeschätzt, ob M_2 die Daten im Verhältnis zur ansteigenden Zahl der Parameter auch signifikant besser wiedergibt. Ist das nicht der Fall, wird die Tendenz zur Überanpassung durch die steigende Anzahl Terme als unverhältnismäßig stark betrachtet und M_1 wird vorgezogen.

3.1.1.2 t-Test

Der F-Test betrachtet das gesamte Modell. Signifikanz bedeutet dann lediglich, dass mindestens eine der Eingangsvariablen einen Zusammenhang beschreibt. Der t-Test kann dagegen angewendet werden, um

die Signifikanz der einzelnen Regressionskoeffizienten zu bewerten. Als Bewertungsmaß dient der t-Wert des Regressionskoeffizienten a_i:

$$t_i = \frac{a_i}{\sigma_{a_i}} \tag{3.1.4}$$

Die Standardabweichung σ_{a_i} beschreibt die Genauigkeit der Schätzung der einzelnen Regressionskoeffizienten. Würde man die Anpassung mit anderen Stichproben aus den Messdaten wiederholen, dann würden auch die a_i unterschiedlich ausfallen und zwar umso stärker, je ungenauer sie aus den Daten ermittelt werden können. σ_{a_i} beschreibt die Streuung dieser Verteilungen der $\{a_i\}$.

Falls a_i keine signifikante Rolle spielt, folgt t_i einer t-Verteilung mit *n-p* Freiheitsgraden. Ist t_i größer als der t-Wert zum Signifikanzniveau α, wird der Regressionsterm beibehalten. Voraussetzung für die korrekte Bestimmung der σ_{a_i} während der Lösung des Gleichungssystems ist Homoskedastizität der Residuen.

3.1.1.3 Mallows Cp

Mallows Cp [60][61] schätzt den mittleren quadratischen Vorhersagefehler eines Regressionsmodells aus dem Fehler auf den Lerndaten ab gemäß

$$C_p = \frac{\sum_{i=1}^{n} (y_i - f(x_i))^2}{\sigma_v^2} - n + 2p \tag{3.1.5}$$

mit

$$\sigma_v = \frac{1}{n} \sum_{i=1}^{n} (y_i - f_v(x_i))^2 \tag{3.1.6}$$

wobei p die im Modell enthaltene Anzahl ausgewählter Parameter ist und σ_v^2 die durch den mittleren quadratischen Fehler abgeschätzte Streuung des vollständigen, alle Parameter enthaltenden Modells. Der erste Term bewertet die Anpassungsgüte und der letzte bestraft die Modellkomplexität. Ein Modell mit $C_p \approx p$ wird in diesem Sinne als adäquat betrachtet. Im Gegensatz zum F-Test muss zum Vergleich zweier Modelle das eine nicht im anderen enthalten sein. Die Berechnung des kompletten Modells kann für große p_{max} und n sehr rechenaufwändig werden.

Mallows Cp ist mit R^2_{corr} (Gleichung 1.6.4) verwandt [62]. Ebenso ist die Leave-One-Out-Kreuzvalidierung asypmtoisch äquivalent zu Methoden wie AIP oder Cp [63].

3.1.1.4 Akaike Informationskriterium

Im Gegensatz zu F, t- und Cp-Statistiken, die aus frequentistischen Überlegungen hergeleitet wurden, betrachtet das Akaike Informationskriterium (AIC) [64] die Anpassungsgüte und Komplexität des Modells aus Sicht der Informationsentropie. Unter gewissen Voraussetzungen korrespondieren diese Überlegungen mit frequentistischen [65].

$$AIC = \ln\left(\sigma^2\right) + 2 \cdot \frac{p}{n} \tag{3.1.7}$$

σ^2ist dabei der mittlere quadratische Fehler. Es wird angenommen, dass keins der zu untersuchenden Modelle die wahren funktionalen Zusammenhänge der Zielfunktion wiedergibt und das Modell, das die Daten am besten mit geringer Komplexität approximiert, erhält das niedrigste AIC.

3.1.1.5 Bayes Informationskriterium

Bei der Herleitung des Bayes-Informationskriteriums (BIC) wird im Gegensatz zu AIC davon ausgegangen, dass das wahre Modell in der Auswahl enthalten ist [66]. Das BIC straft zusätzliche Parameter schärfer als das AIC und tendiert daher weniger stark zur Überanpassung, umgekehrt besteht hier eher die Gefahr der Unteranpassung.

$$BIC = \ln\left(\sigma^2\right) + \ln(n) \cdot \frac{p}{n} \tag{3.1.8}$$

3.1.1.6 Resampling-Verfahren

Falls nur wenige Messungen n zur Verfügung stehen, ist die Schätzung von statistischen Kenngrößen, etwa Mittelwert oder Varianz der Fehlerverteilung, ungenau. Dies kann durch Resampling-Strategien verbessert werden, indem mehrfache Auswertungen auf unterschiedlichen Teilmengen der Daten wiederholt werden. Zur Fehlerabschätzung bei Regressionsmodellen werden dazu neben der Kreuzvalidierung [63, 67] oft Bootstrap-Methoden [68, 69] verwendet:

Sei $\bar{s}$ der aus der Verteilung $S = (s_1, ..., s_N)$ berechnete Mittelwert als Schätzung des wahren Mittelwerts m der zugrundeliegenden Verteilung. Beim Bootstrapping werden aus dieser Verteilung wiederholt zufällig B Bootstrap-Stichproben $S_b = (s_{b1}, ..., s_{bN})$, $b = 1, ..., B$ mit Zurücklegen gezogen [70, 71]. Elemente s_i können in den S_b also mehrfach auftreten. Dadurch ist die Bootstrap-Verteilung ähnlich der zugrundeliegenden abzuschätzenden Verteilung. Für jede Bootstrap-Probe S_b wird der Mittelwert $\bar{s}_b$ berechnet. Für große B sind die $\bar{s}_b$ dann normalverteilt und verhalten sich zum Mittelwert $\bar{s}$ analog wie $\bar{s}$ zum wahren Mittelwert m . Dadurch kann die Genauigkeit von $\bar{s}$ abgeschätzt werden.

Neben diesen weitläufig angewendeten Kriterien existieren weitere ähnliche Ansätze zur Abschätzung der Modellqualität, z. B. das Risk Inflation Criterion [72] oder das Covariance Inflation Criterion [73] oder Ansätze, die auf generalisierten Freiheitsgraden beruhen [74].

3.1.2 Modellauswahlverfahren

Es existiert eine Vielzahl an Methoden zur Bewertung und Auswahl von Modellen, um eine effiziente Suche nach einem geeigneten Modell zu ermöglichen [75, 76, 77, 78]. Bei den folgenden Methoden wird nach Modellen gesucht, die eine dem Problem angemessene Menge von Termen enthalten. Dazu werden verschiedene aus unterschiedlichen Termkombinationen zusammengesetzte Modelle an die Lerndaten angepasst. Das Modell mit dem höchsten R^2 oder R^2_{corr} bzw. niedrigsten MSE wird ausgewählt, falls seine Anpassungsgenauigkeit im F-Test als signifikant betrachtet wird. Allerdings ist die pauschale Vorgabe eines kritischen F-Werts zu einem Signifikanzniveaus α problematisch. Durch kleine Werte von F werden große Modelle bevorzugt. Falls nur wenige Messungen zur Verfügung stehen, ist die Schätzung von statistischen Kenngrößen, etwa Mittelwert oder Varianz der Fehlerverteilung, ungenau. Dies kann durch Resampling-Strategien verbessert werden, indem mehrfache Auswertungen auf unterschiedlichen Teilmengen der Daten wiederholt werden. Zur Fehlerabschätzung bei Regressionsmodellen werden dazu neben der Kreuzvalidierung [63, 67] oft Bootstrap-Methoden [68, 69] verwendet.

Bei Problemen mit hohem n/p hängt die adäquate Wahl von F von der Anzahl auszuwählender Terme ab [53]. Als Folge können Abschätzungen zu fest vorgegebenem F bei einer hohen Anzahl Eingangsparameter zu überaus optimistischen Schätzungen führen, so dass die tatsächliche Signifikanz eines Modells überbewertet wird und das Modell überanpasst [79]. In der Literatur wurden daher Methoden zur Ermittlung geeigneter Signifikanzkriterien vorgestellt [80, 81, 82, 83].

Falls die Anzahl der Auswahlterme p_{max} gering ist, können alle möglichen Modelle aus der Menge der Auswahlterme gebildet und überprüft werden (Best Subset Selection). Für die hier betrachteten typischen Werte von $p_{max} > 1000$ würde die Berechnung der $2^{p_{max}} - 1$ Modelle aber viel zu lange dauern. Daher kommen nur Methoden in Frage, die die Menge aller möglichen Modelle möglichst zielstrebig absuchen, so dass nur eine übersichtliche Teilmenge untersucht werden muss. Dies ist Ziel der schrittweisen Regression (Stepwise Regression [84]). Verfahren, die in einem Schritt jeweils ein bezüglich eines Kriteriums bestes Element hinzufügen, zählen zu den Greedy Algorithms [85]. Ein grundlegend anderer Ansatz ist die Auswahl mittels genetischer Algorithmen [86], deren Rechenaufwand aber auch nur für relativ niedrige p_{max} vertretbar ist.

Regressionsverfahren ermitteln Polynome, die aus statistischer Sicht gut zum Problem passen. Bei der Interpretation der Koeffizienten ist aber Vorsicht geboten: sie alleine machen außer in einfachen niedrigdimensionalen Fällen keine direkt interpretierbare Aussage über das Verhalten eines Parameters. In

hochdimensionalen Fällen sollten nicht die Koeffizienten interpretiert werden, sondern die Funktionsverläufe des Modells um einen gegebenen Arbeitspunkt $(x_1, ..., x_{i-1}, x_{i+1}, ..., x_m)$, wobei x_i innerhalb eines interessierenden Intervalls variiert wird.

3.1.2.1 Vorwärtsselektion

Vorwärtsselektion (Forward selection) ist eine Variante der schrittweisen Regression. Sie verfolgt den Ansatz, ausgehend von einem nur eine Konstante enthaltenen Modell nacheinander jeweils diejenige noch nicht im Modell enthaltene Eingangsgröße hinzuzufügen, durch die das Modell in jedem Schritt eine möglichst große Varianz der Zielgröße erklären kann [87, 88]. Dieses Verfahren wird fortgesetzt, bis der F-Wert der besten hinzugefügten Eingangsgröße unter eine vorgegebene Schranke fällt [84, 89].

Zur Bewertung der Modelle werden Informationskriterien verwendet, die eine hohe Anzahl an Eingangsgrößen bestrafen, so dass Modelle mit wenigen Eingangsgrößen bei gleicher Varianz solchen mit mehr Eingangsgrößen bevorzugt werden. Anstatt F-Tests können auch andere Informationskriterien verwendet werden, etwa das korrigierte Bestimmtheitsmaß (Gleichung 1.6.4), das Akaike-Informationskriterium, das Bayes-Informationskriterium oder Mallows Cp. Ihnen ist gemeinsam, dass sie eine höhere Anzahl an Parametern strafend berücksichtigen, so dass Modelle mit möglichst wenig Parametern bevorzugt werden, solange sie nicht signifikant schlechtere Fehler liefern als komplexere Modelle. Vorwärtsselektionsmethoden werden vielseitig in der Regressionsanalyse angewendet [85]. Manche Methoden erstellen zuerst ein komplexes Modell mittels Vorwärtsselektion, um dann Methoden darauf anzuwenden, die das Modell wieder vereinfachen, z. B. Shrinkage-Methoden [90] (Abschnitt 3.1.3).

3.1.2.2 Rückwärtselimination

Bei der Rückwärtselimination (Backward elimination) wird mit einem Modell begonnen, das alle Eingangsvariablen enthält. Das Modell wird schrittweise vereinfacht, indem jede enthaltene Variable auf ihre Signifikanz geprüft und diejenige mit dem geringsten F-Wert entfernt wird [91]. Diese Schritte werden fortgesetzt, bis das Minimum der F-Werte der enthaltenen Variablen eine vorgegebene Schranke erreicht.

Bei einer hohen Anzahl an auszuwählenden Parametern und großen Datenmengen eignet sich eine Vorwärtsselektion, da der Rechenaufwand für das Auswerten von Modellen, die fast alle Parameter enthalten, sonst sehr hoch werden kann. Wenn die Anzahl der Parameter nicht sehr groß ist und zu erwarten ist, dass zur Ermittlung eines angemessenen Modells lediglich einige wenige Parameter entfernt werden müssen, eignet sich eine Rückwärtselimination.

Für $n < p$ ist das Gleichungssystem unterbestimmt und damit nicht eindeutig lösbar. Es kann aber durch Zusatzbedingungen eindeutig lösbar gemacht werden, etwa durch die Forderung, dass der Koeffizientenvektor a minimal bezüglich einer vorgegebenen Norm ist.

3.1.2.3 Bidirektionale Regression

Während das Modell an Komplexität gewinnt, ändert sich die Signifikanz der einzelnen Variablen. Solche, die zu einem früheren Zeitpunkt relevant waren, können in einem komplexeren Modell durch eine Kombination anderer Variablen ersetzt werden, die den funktionalen Zusammenhang besser wiedergeben. Um diese zu entfernen, werden bei der bidirektionalen Regression Vorwärtsselektion und Rückwärtselimination kombiniert [92]. Zuerst wird eine Vorwärtsregression durchgeführt und nach derern Abschluss eine Rückwärtselimination. Vorwärts- und Rückwärtsregression können auch derart kombiniert werden, so dass nach jedem Vorwärtsschritt mithilfe von Signifikanzkriterien überprüft wird, ob jede bereits im Modell enthaltene Variable durch eine signifikantere ersetzt werden kann.

3.1.3 Shrinkage-Methoden

Shrinkage-Methoden in der Regression versuchen, Überanpassung zu minimieren, indem die zu schätzenden Koeffizienten betragsmäßig klein gehalten werden [93]. Als Folge nimmt die Varianz des Modells zugunsten eines geringeren Bias-Fehlers ab [94].

Bei der Koeffizientenberechnung können zusätzliche Bedingungen gestellt werden, z. B. durch Beschränkung der Summe der Quadrate der Koeffizienten durch eine Konstante t. Dadurch werden weniger signifikante Koeffizienten gegen Null gedrückt (Ridge Regression [95]). Die Lösung des Gleichungssystems wird dann als Minimierung unter Nebenbedingungen durch einen Lagrange-Multiplikator $\lambda \geq 0$ formuliert:

$$\begin{aligned} (y-Aa)^T (y-Aa) &= \min, \quad \sum_{i=1}^{p} a_i^2 \leq t \\ \Leftrightarrow \quad \tfrac{1}{n} (y-Aa)^T (y-Aa) + \lambda \sum_{i=1}^{p} a_i^2 &= \min \end{aligned} \tag{3.1.9}$$

λ ist hierbei ein vom Benutzer vorgegebener Parameter. Der Grenzfall $\lambda = 0$ beschreibt die Least Squares-Lösung. Für $\lambda \to \infty$ gehen alle Koeffizienten gegen Null. Somit wird durch λ der Kompromiss zwischen Verzerrung und Varianz (Abschnitt 1.7.1) gesteuert. Im Vergleich zur Ordinary Least Squares-Methode liefert diese Nebenbedingung numerisch stabilere Ergebnisse, wenn die Vektoren zweier Eingangsparameter fast linear abhängig sind (Multikollinearität) und $A^T A$ dadurch fast singulär wird. Es existieren diverse Methoden zur automatischen Wahl des Ridge-Parameters λ bzw. t [96].

Der Ansatz der Ridge-Regression erlaubt im Gegensatz zur Ordinary Least Squares-Methode auch die Berechnung des Gleichungssystems für den Fall $n < p$, was bei einer Backward elimination auftreten kann.

Durch Beschränking der Summe des Absolutbetrags der Koeffizienten beim Least Absolute Shrinka-

ge Operator (LASSO [97]) können Koeffizienten sogar exakt Null werden, so dass eine Variablenselektion stattfindet.

$$\begin{aligned} (y-Aa)^T (y-Aa) &= \min, \quad \sum_{i=1}^{p} |a_i| \leq t \\ \Leftrightarrow \quad \tfrac{1}{n} (y-Aa)^T (y-Aa) + \lambda \sum_{i=1}^{p} |a_i| &= \min \end{aligned} \tag{3.1.10}$$

In der Elastic Net Regularization [98] werden die Beschränkungen der Quadrate und Absolutbeträge kombiniert. Als Ausgangspunkt für Shrinkage-Methoden kann das vollständige Modell oder ein anderes zu komplexes Modell dienen. Shrinkage-Methoden können auch mit schrittweisen Methoden kombiniert werden, z. B. einer Forward Selection zur Erstellung eines Ausgangsmodells, bis ein Abbruchkriterium erfüllt ist und anschließender Elimination von Termen [99].

Durch die Bedingung, die Summe der Koeffizientenbeträge oder ihrer Quadrate zu beschränken, wird das Problem normierungsabhängig [100]. Üblicherweise wird Vergleichbarkeit hergerstellt, indem die x_i durch Zentrierung um deren Mittelwert und Normierung auf Standardabweichung standardisiert werden. Wenn die Verteilung der x_i aber stark von einer Normalverteilung abweicht, ist diese Vergleichbarkeit nicht mehr gegeben. In den in dieser Arbeit untersuchten Datensätzen finden sich oft multimodale Verteilungen der x_i, die sich aus den Gruppierungen unterschiedlicher Produktanforderungen ergeben.

Wenn die grundlegenden funktionalen Zusammenhänge der zu approximierenden Funktion linear sind, ist ein Vergleich der Koeffizienten adäquat skalierter Eingangsparameter-Vektoren als globales Maß ihres Beitrags zur Regression möglich, bei nichtlinearen Zusammenhängen dagegen nicht. So wirken sich beispielsweise im Regressionsansatz $f(x) = a_1 \cdot x + a_2 \cdot x^2$ die beiden Terme nicht für alle x gleich stark aus. Für $x < 1$ überwiegt der lineare Term zum Koeffizienten a_1, für $x > 1$ der quadratische zu a_2.

3.1.4 Boosting

Boosting ist eine Strategie, die auf unterschiedliche Klassen von statistischen Modellen angewendet werden kann, unter anderem auf Polynomregression [101, 102]. Auf Polynome kann sie in Form einer Stagewise Regression [103] angewendet werden: Ausgehend von einem einfachen Modell mit hohem Bias-Fehler und niedriger Varianz wird der Residuenvektor nochmals durch das Modell angepasst und mit einer zu wählenden Gewichtung auf das Ausgangsmodell addiert. Dieser Vorgang wird bis zur Erfüllung einer Abbruchbedingung wiederholt, so dass die Residuen des Vorgängermodells jeweils schrittweise verringert werden [104, 105, 106, 44]. Die Komplexität des Modells steigt dabei langsamer als bei einer schrittweisen Regression, da die hinzugefügten Terme bereits im Modell enthalten sein können.

3.2 Strategie der Polynomanpassung

Um eine Überanpassung zu vermeiden, sollte die Anzahl der zu bestimmenden Parameter des Modells möglichst gering gehalten werden. Polynome mit zu niedriger Ordnung sind aber nicht in der Lage, komplexe funktionale Zusammenhänge zufriedenstellend genau zu reproduzieren. Ziel der adaptiven Entwicklung ist es, ein Polynom zu finden, das nur geeignete Terme aus Produkten von Potenzen der Eingangsgrößen enthält.

Im Folgenden wird ein neues eigenes Verfahren vorgestellt, das ein Polynommodell schrittweise in mehreren Stufen erweitert und flexibel wieder vereinfacht, bis ein Modell erreicht ist, das die grundlegenden Zusammenhänge der zu modellierenden Daten mit angemessener Komplexität wiedergibt, ohne dabei durch zufällige Schwankungen stark verfälscht zu werden. Zur Funktionsapproximation werden ganzzahlige Potenzen der Eingangsgrößen und Produkte daraus benutzt, die Methode kann aber auch auf nicht-ganzzahlige Potenzen wie Fractional Polynomials [107] angewendet werden.

Im Unterschied zur schrittweisen Regression wird die Güte des Modells nicht durch die in Abschnitt 3.1.1 vorgestellten Signifikanztests bewertet und kommt daher ohne die ihnen zugrundeliegenden zusätzlichen Annahmen aus. Anstatt den Fehler auf den Lerndaten bezüglich der Mächtigkeit des Datensatzes und der im Modell enthaltenen Terme zu gewichten, findet die Modellauswahl mittels Kreuzvalidierung statt. Da eine höhere Anzahl an Termen nicht notwendigerweise zu einer Überanpassung führt, wird sie im Gegensatz zur üblichen schrittweisen Regression nicht strafend in der Bewertung berücksichtigt und das Verfahren endet nicht, wenn eine gewisse Signifikanzschranke erreicht wurde. Damit wird dem Problem entgegengewirkt, dass die Modellentwicklung frühzeitig abgebrochen wird, obwohl ein komplexeres Modell zu einem späteren Zeitpunkt die Daten deutlich besser wieder geben könnte. Eine Mehrfach-Kreuzvalidierung benötigt zwar mehr Rechenleistung als eine einzelne Auswertung auf dem gesamten Datensatz und anschließende Modellbewertung mit den in Abschnitt 3.1.1 vorgestellten Gütekriterien. Der gemittelte Vorhersagefehler ist aber ein direktes Maß für die Vorhersagegüte des Modells.

Algorithmus 3.1 zeigt den Pseudocode dieser adaptiven Polynomentwicklung.

Algorithmus 3.1 *Pseudocode der adaptiven Polynom-Entwicklung.*

Erstelle D_{test} und D_{train} aus D (Gl. 1.6.5)
$i = 0$
$M^{(0)} = 1$
$S = \left\{ \prod_{j=1}^{n} x_j^{p_j} \mid \forall j : p_j \leq p \text{ und } \sum_{j=1}^{n} p_j = q_{max} \right\}$
iteriere, bis Abbruchkriterium erfüllt

ERWEITERN: für $j = 1, ..., \delta^+$:
$i = i + 1$
für alle $s \in S \setminus M^{(i-1)}$:
$M_c^{(i)} = \{M^{(i-1)} \cup \mathrm{s}\}$
fit $\left\{ M_c^{(i)} \right\}$ an D_{train}
berechne $MSE_{test}\left(M_c^{(i)} \right)$
falls $M_c^{(i)} \neq M^{(i-2)}$:
$M^{(i)} := M_c^{(i)}$ mit $MSE_{test}\left(M_c^{(i)} \right) = \min$
sonst:
$M^{(i)} = M^{(i-1)}$
Erstelle D_{test} und D_{train} aus D
beende *ERWEITERN*

REDUZIEREN: für $j = 1, ..., \delta^-$:
$i = i + 1$
für alle $s \in M^{(i-1)}$:
$M_c^{(i)} = \{M^{(i-1)} \setminus \mathrm{s}\}$
fit $\left\{ M_c^{(i)} \right\}$ an D_{train}
berechne $MSE_{test}\left(M_c^{(i)} \right)$
falls $M_c^{(i)} \neq M^{(i-2)}$:
$M^{(i)} := M_c^{(i)}$ mit $MSE_{test}\left(M_c^{(i)} \right) = \min$
sonst:
$M^{(i)} = M^{(i-1)}$
Erstelle D_{test} und D_{train} aus D
beende *REDUZIEREN*

falls $MSE_{test}\left(M^{(i)} \right) > MSE_{test}\left(M_c^{(i-1)} \right)$:
REDUZIEREN
sonst falls $M^{(i)} \in \{M^{(1)}, ..., M^{(i-1)}\}$:
Erstelle D_{test} und D_{train} aus D

fit $M^{(i)}$ an D

Um das Modell an die Daten anzupassen, stehen alle Kombinationen aus den Eingangsparametern

x_1, ..., x_m bis zur maximalen kombinierten Ordnung q_{max}

$$S = \left\{ \prod_{i=1}^{m} x_i^{p_i} \mid \forall i : \, p_i \in \mathbb{N}_0 \leq p \text{ und } \sum_{i=1}^{m} p_i \leq q_{max} \right\} \tag{3.2.1}$$

zur Verfügung. Mit Termen der kombinierten Ordnung $q_{max} = 3$ lassen sich somit lineare Zusammenhänge x_i, Krümmungen x_i^2, Wendepunkte x_i^3 und Wechselwirkungen $\left\{ x_i x_j, x_i x_j x_k, x_i^2 x_j \mid i \neq j \neq k \right\}$ darstellen. Der betrachtete Raum der Polynome H ist

$$H = f(x) = \sum_{c \in S} a_c \cdot c \mid a_c \in \mathbb{R} \tag{3.2.2}$$

Ein Modell ist eine Untermenge von S, das den Raum der Fit-Funktionen determiniert:

$$H_M := f_M(x) = \sum_{c \in M} a_c \cdot c \mid a_c \in \mathbb{R} \text{ und } M \subset S \tag{3.2.3}$$

mit den Regressionskoeffizienten a_c.

3.3 Erweiterungsschritte

Das Ausgangsmodell ist konstant, $M^{(0)} = 1$. Aus dem Ausgangsmodell werden Kandidatenmodelle erstellt, indem je ein Auswahlterm $s \in S$ hinzugefügt wird. Dadurch entstehen $|S|$ Kandidatenmodelle

$$\left\{ M_{c1}^{(0)}, ..., M_{c|S|}^{(0)} \right\} \tag{3.3.1}$$

Jedes dieser Modelle

$$\left\{ f_{Mci}^{(0)} \right\} = konst_{ci}^{(0)} + a_{ci}^{(0)} \cdot s_{ci}, \, s_{ci} \in S \tag{3.3.2}$$

wird an D_{train} angepasst und der Vorhersagefehler für D_{test} berechnet. Das Kandidatenmodell $M_{ci}^{(0)}$ mit dem geringsten Vorhersagefehler wird als Ausgangsmodell für den nächsten Iterationsschritt N übernommen:

$$M^{(N)} := M_{ci}^{(N-1)} \text{ mit } MSE_{test} \left(M_{ci}^{(N-1)} \right) = \min \tag{3.3.3}$$

Im folgenden Iterationsschritt werden neue Kandidatenmodelle erstellt, indem je einer der bisher noch nicht im Modell enthaltenden Auswahlterme hinzugefügt wird:

$$\left\{M_{ci}^{(N)}\right\} = M^{(N)} \cup s_c \forall s_c \in S \setminus M^{(N)} \tag{3.3.4}$$

Durch iteratives Fortsetzen dieses Verfahrens wird die Komplexität des Modells schrittweise erhöht. Dadurch können die Zieldaten immer besser approximiert werden, bis weiteres Hinzufügen von Termen schließlich keine Verbesserung mehr bringt. Zusätzlich werden aus dem Modell in regelmäßigen Abständen Terme mit geringer Relevanz entfernt. Dies wird in Abschnitt 3.5 ausführlicher erläutert.

Alternativ könnte auch ein komplexes Ausgangsmodell schrittweise vereinfacht werden. Da aber ein vollständiges Polynommodell hohen Grades eine sehr hohe Anzahl Terme aufweist (siehe Abs. 7.1.2), besteht die Gefahr der Überanpassung oder das Problem wird für $p > n$ nicht mehr eindeutig lösbar.

3.4 Fehlerbewertung

Zur Bewertung der Anpassungsgüte des Modells wird der Fehler auf den Testdaten herangezogen, hier der Testfehler MSE_{test}. Da die least-squares Lösung des Gleichungssystems das globale Optimum darstellt, kann eine Kreuzvalidierung über den gesamten vorhandenen Datensatz durchgeführt werden, ohne dass die Gefahr besteht, dass in den einzelnen Validierungsschritten Nebenminima gefunden werden, die untereinander nicht zum Vergleich geeignet sind. Eine durchgehende Trennung von Test- und Trainingsdaten ist daher nicht nötig.

Diese Überprüfung des Modells über den gesamten Datensatz im Gegensatz zu einer festgehaltenen disjunkten Trainings- und Testdatenmenge ist insbesondere bei dünnen und stark verrauschten Daten von Vorteil, da in diesem Fall die Annahme gleichmäßig über beide Datenmengen verteilter Information und Fehlerstreuung im Allgemeinen kaum erfüllt ist. Folglich wäre der Fehler auf einem einzigen Testdatensatz nicht unbedingt repräsentativ. Bei der Kreuzvalidierung sieht jedes Modell auf den einzelnen Testdatensätzen zwar ähnliche, aber unterschiedliche Fehlerlandschaften.

Der Algorithmus kann zur Bewertung entweder den über die einzelnen Kreuzvalidierungsschritte gemittelten Testfehler oder den größten Einzelfehler aus allen Schritten verwenden. Die Wahl des größten Testfehlers hat eine hohe Sensibilität gegenüber ungleichmäßigen Verteilungen der Daten zur Folge. Gerade bei einer hohen Anzahl an Eingangsparametern und verhältnismäßig wenig Messdaten kann es bei einer zufälligen Verteilung der Messdaten auf die Trainings- und Testmengen aber vorkommen, dass die Menge der Messdaten eines einzelnen Testdatensatzes nicht hinreichend gleichmäßig über die Gesamtmenge der Messdaten verteilt ist. Damit ist diese Testdatenmenge nicht repräsentativ und die Annahme, dass der Fehler auf den Testdaten die Verallgemeinerungsfähigkeit des Modells repräsentiert, nicht mehr erfüllt. Je nachdem ob diese Teilmenge hauptsächlich in einem Gebiet liegt, die vom Modell gut oder schlecht approximiert wird, kann somit ein unverhältnismäßig niedriger oder hoher Fehler entstehen.

Ebenso beschränkt eine unräpräsentative Verteilung auf die Testdaten die Anpassungsfähigkeit des Modells. In solchen Fällen führt die Wahl des größten Testfehlers dazu, dass sich das Modell aufgrund der starken Fehlerschwankungen in eine Richtung entwickelt, die insgesamt kaum eine Verbesserung bringt und nach relativ wenigen Iterationsschritten nicht weiter verbessert werden kann.

Die Wahl des mittleren Testfehlers schwächt solche Ausreißer dagegen ab und erlaubt dem Modell, sich weiter zu verfeinern. Je höher die Anzahl der Fehlerberechnungen bei der Kreuzvalidierung ist, desto höher ist ihre statistische Aussagekraft. Eine k-fach Kreuzvalidierung mit hohem k erzeugt zwar viele Fehlermessungen, aber auch eine hohe Varianz (Abschnitt 1.7.1). Alternativ können c-mal nacheinander mehrere k-fach Kreuzvalidierungen durchgeführt werden, zwischen denen die Daten zufällig neuverteilt werden [108]. Dadurch können maximal $n \cdot k$ Fehlerwerte generiert werden, auf denen dann z. B. t-Test durchgeführt werden kann, um zu bewerten, wie signifikant sich zwei Modellkandidaten unterscheiden. Der dadurch erzeugte c-fache Rechenaufwand ist bei den meisten hier betrachteten Problemen aber nicht vertretbar.

Abbildung 3.2 zeigt den Vergleich zweier Anpassungen, bei denen die Modelle mit einer 4-fach Kreuzvalidierung einmal durch den gemittelten und einmal durch den schlechtesten MSE_{test} bewertet wurden. Verwendet wurde ein HIC-Datensatz mit 992 stark verrauschten Messungen bei 19 Eingangsgrößen, aus denen Auswahlterme bis zur 3. Ordnung gebildet wurden (1540 Terme). In jedem Anpassungsschritt wurden zunächst zwei Terme hinzugefügt und dann mindestens ein Term entfernt. Erläuterungen zur Entfernung von Termen folgen in Abschnitt 3.5.

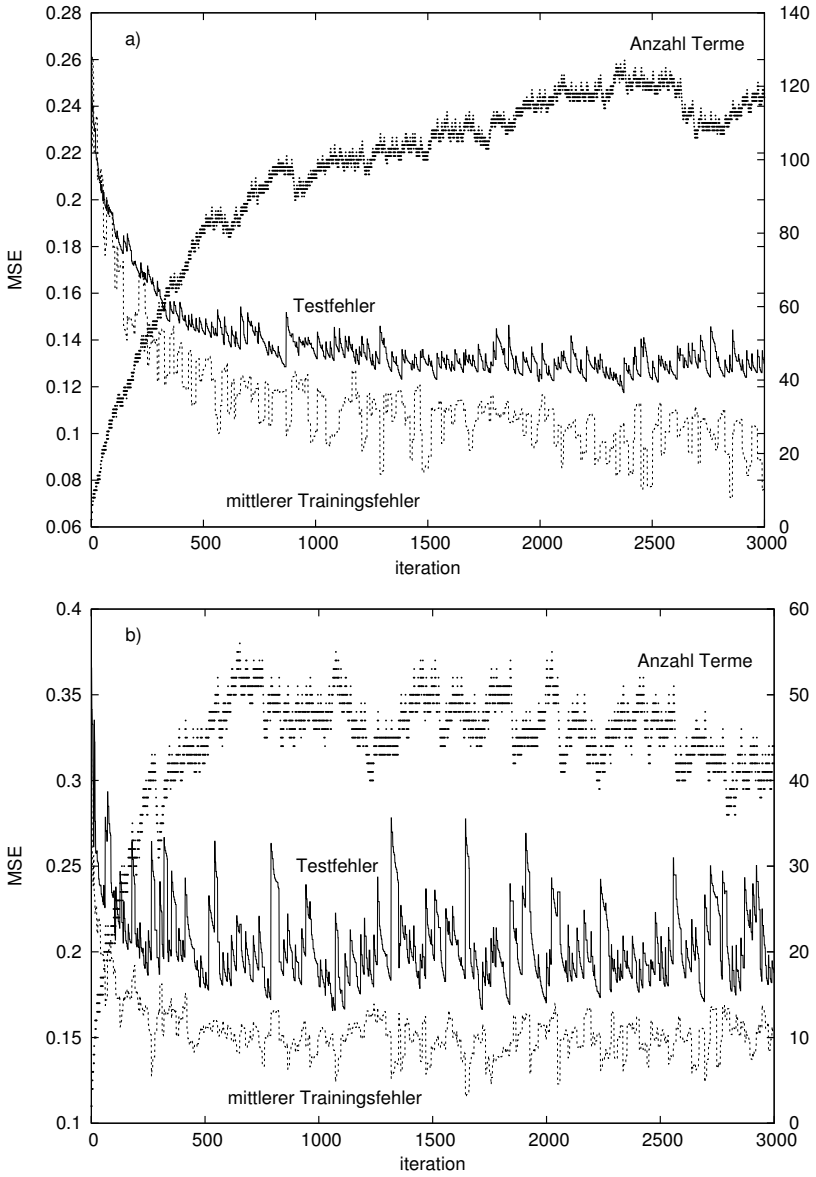


Abbildung 3.2: Iterationsverlauf für a) den gemittelten und b) den schlechtesten MSE_{test} zur Bewertung der Anpassungsgüte an einen HIC-Datensatz. Im Fall a) entsteht ein Modell höherer Komplexität (ausgedrückt durch die Anzahl Terme), das die Trainingsdaten besser abbildet als Fall b) und gleichzeitig eine bessere Verallgemeinerungsfähigkeit auf den Testdaten erzielt.

Die Sprünge im Fehlerverlauf sind Stellen, an denen Test- und Trainingsdaten zufällig permutiert wurden (siehe Abschnitt 3.8). Dabei sieht das Modell nicht mehr die selbe Fehlerlandschaft, so dass der Fehler sich üblicherweise stärker ändert als in den einzelnen Erweiterungs- und Vereinfachungsschritten.

Zum Vergleich ist bei beiden Modellen der mittlere Fehler auf den Trainingsdaten aufgetragen. Nach etwa 1500 (Modell a)) bzw. 600 (Modell b)) Iterationsschritten ändern sich Testfehler und Komplexi-

tät der Modelle, wiedergegeben durch die Anzahl verwendeter Terme, nicht mehr wesentlich. Modell b), das den schlechtesten MSE_{test} als Bewertungskriterium verwendet, enthält dann nur etwas mehr als ein Drittel der Terme des Modells a), das den mittleren MSE_{test} verwendet. Im Anschluss wurden die Koeffizienten der beiden Modelle mittels einer 10-fach Kreuzvalidierung überprüft. Modell a) lieferte dabei einen MSE_{test} von 0,151 und ein Bestimmtheitsmaß auf dem gesamten Datensatz von $R^2 = 0,68$ und Modell b) $MSE_{test} = 0.186$ und $R^2 = 0,51$. Damit ist Modell a) in der Lage, die Komplexität der Zielfunktion genauer wiederzugeben und gleichzeitig weniger überanzupassen als Modell b).

Das bei den HIC-Daten erreichte relativ niedrige Bestimmtheitsmaß deutet auf eine schlechte Wiedergabe der Daten durch das Modell hin. Dies liegt aber unter anderem an der starken Streuung der Messwerte zu gleichen Eingangsparametern, die durch kein funktionales Modell vollständig eliminiert werden kann. Dies wird in der Zusammenfassung (Kapitel 6) in einem Vergleich mit grundsätzlich verschiedenen Modellansätzen verdeutlicht. Trotz des niedrigen Bestimmtheitsmaßes haben die Modelle in der Praxis geholfen, die Prozessführung bezüglich HIC zu verbessern.

Die mindestens notwendige Anzahl k der Trainings- und Testdatensätze hängt von der Anzahl n der Messdaten, der Anzahl m der Eingangsparameter, der Komplexität des abzubildenden Zusammenhangs und von den Fehlern in Eingangs- und Ausgangsgröße ab. Bei der minimalen Anzahl von $k = 2$ Kreuzvalidierungsschritten wird nur die Hälfte der Daten zum Trainieren benutzt. In diesem Fall muss sichergestellt sein, dass diese Datenmenge auch ausreichend Information enthält, um das Problem zufriedenstellend abbilden zu können. Bei niedrigem n ist das fraglich.

Wenn das Modell bei der Kreuzvalidierung auf keinem der Testdatensätze überanpasst, d. h. wenn der Fehler auf den Testdaten hinreichend klein bleibt, ist es als robust zu betrachten. Nach Beendigung der Iteration können die Koeffizienten des abschließenden Modells auf den gesamten Datensatz angepasst werden. Die somit bewirkte Erhöhung der Trainingsdaten bei gleicher Anzahl an Modelltermen stabilisiert die Anpassung gegenüber Überanpassung und berücksichtigt die Information aller Messdaten.

3.5 Vereinfachung der Termstruktur

Im Verlauf der Iteration kann es vorkommen, dass bereits im Modell enthaltene Terme an Bedeutung verlieren, da die von ihnen approximierten Zusammenhänge durch eine Kombination später hinzugefügter Terme exakter modelliert werden können. Dies soll am Beispiel einer adaptiven Anpassung an Funktionswerte der Funktion $f(x) = x^3 - x^2$ im Intervall $[0; 0,5]$ erläutert werden (Abbildung 3.3).

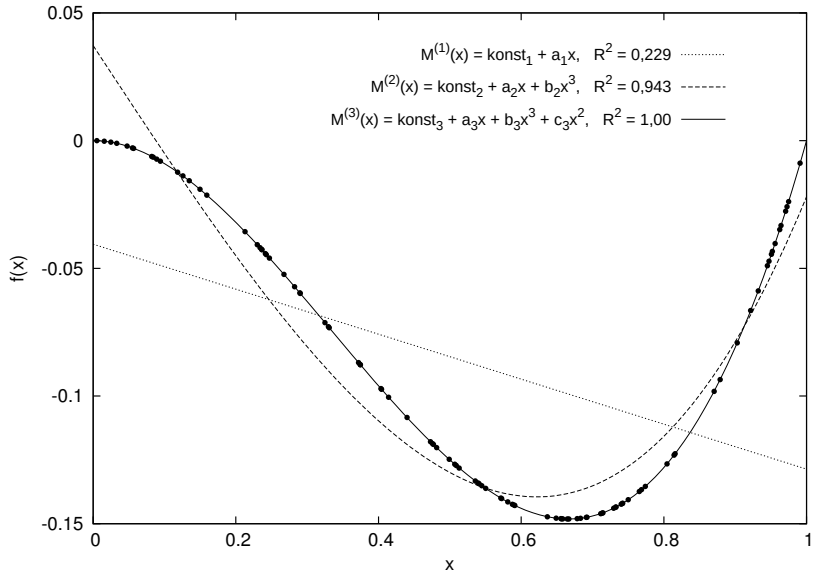


Abbildung 3.3: Schrittweise Anpassung eines adaptiven Polynommodells an Datenpunkte der Funktion $f(x) = x^3 - x^2$. Ausgehend von der konstanten Funktion $M^{(0)}$ (nicht dargestellt) wird als beste Least Squares-Anpassung zunächst das lineare Modell $M^{(1)}$ ausgewählt und im Folgeschritt zur kubischen Funktion $M^{(1)}$ erweitert. Im letzten Schritt wird durch Hinzunahme eines quadratischen Terms die Funktion $M^{(3)}$ generiert, durch welche die Funktion $f(x)$ exakt wiedergegeben werden kann.

Das konstante Ausgangsmodell wird im ersten Iterationsschritt probeweise um die Terme x, x^2 und x^3 erweitert. Da x die in dem Intervall monoton fallende Funktion besser approximiert als x^2 oder x^3 alleine, erhält das Kandidatenmodell

$$f^{(0)}_{Mc1}(x) = konst^{(0)}_{c1} + a^{(0)}_{c1} \cdot x$$

einen niedrigeren Vorhersagefehler als die beiden anderen Kandidaten

$$f^{(0)}_{Mc2}(x) = konst^{(0)}_{c2} + a^{(0)}_{c2} \cdot x^2 \text{ and } f^{(0)}_{Mc3}(x) = konst^{(0)}_{c3} + a^{(0)}_{c3} \cdot x^3$$

Im nächsten Schritt wird $M^{(1)} = M^{(0)}_{c1}$ am stärksten verbessert, wenn der Term x^3 hinzugefügt wird:

$$f^{(2)}_{M}(x) = konst^{(1)}_{2} + a^{(1)}_{21} \cdot x + a^{(1)}_{22} \cdot x^3$$

Im Folgeschritt wird das Modell durch die Erweiterung um x^2 exakt und es ergibt sich

$$f^{(3)}_{M}(x) = konst_3 + f(x) + a_3 x$$

Durch Entfernen je eines Terms aus $M_3(x)$ und Neubewertung der Anpassungsgüte wird ersichtlich, dass sich der Anpassungsfehler bei Entfernen der Konstante und des linearen Terms x nicht verschlechtert, während er bei Entfernen des quadratischen oder kubischen Terms deutlich schlechter wird. Die Konstante und der lineare Term sind damit überflüssig und würden bei verrauschten Daten die Gefahr der Überanpassung erhöhen.

Das obige Beispiel veranschaulicht, dass das Hinzufügen jeweils eines Terms pro Schritt nicht immer ausreicht, um die Zusammenhänge zwischen Eingangsparametern und Zielfunktion abzubilden. Statt dessen wird zunächst ein Term gewählt, der den besten Kompromiss aus Komplexität und Überanpassung darstellt. Durch Hinzufügen weiterer Terme mit den entsprechenden Eingangsparametern kann das Modell die Zusammenhänge sukzessive besser wiedergeben. Diese Terme dienen als Ausgangspunkt, um die zur möglichst exakten Abbildung benötigten Terme überhaupt erst zu finden. Dabei können sie selbst aber im weiteren Verlauf der Iteration überflüssig werden und sollten entfernt werden, da sie sonst die Gefahr der Überanpassung erhöhen. Als Folge kann sich der Fehler auf den Testdaten im nächsten Iterationsschritt sogar verschlechtern. In diesem Fall sollte das Modell vereinfacht werden. Dazu wird vom bestehenden Modell der Reihe nach jeweils ein Term probeweise entfernt und die Modellkoeffizienten neu angepasst:

$$\left\{M_{ci}^{(N+1)}\right\} = M^{(N)} \setminus s_M \forall s_M \in M^{(N)} \tag{3.5.1}$$

Der Term, ohne den sich das Modell am wenigsten verschlechtert oder am meisten verbessert, wird dann entfernt:

$$M^{(N+1)} := M_{ci}^{(N)} \text{ mit } MSE_{test}\left(M_{ci}^{(N)}\right) = \min \tag{3.5.2}$$

Eine Verschlechterung des Testfehlers von einem Iterationsschritt zum nächsten ist keine notwendige Bedingung dafür, dass das Modell zu komplex ist. Daher sollten unabhängig von der Fehlerentwicklung in regelmäßigen Abständen die am wenigsten signifikanten Terme entfernt werden. Selbst wenn sich das Modell dadurch zunächst nicht oder nicht nennenswert verbessert, wird es dennoch ermöglicht, die Terme im nächsten Schritt durch geeignetere auszutauschen oder sie, falls nötig, wieder hinzuzufügen. Falls der Testfehler dieses vereinfachten Modells besser ist als der des Vorgängermodells, wird nochmals der am wenigsten signifikante Term ermittelt und weggelassen. Das wird so lange wiederholt, bis der Testfehler größer oder gleich dem des Vorgängermodells zu Beginn der Vereinfachungsphase ist. Dieses Abspecken bei Bedarf erlaubt es dem Algorithmus, die Termstruktur ausgiebig zu optimieren, da die zu ersetzende Menge an redundant gewordenen Termen größer sein kann als die vorgegebene Mindestzahl an Ersetzungsschritten und deren Ersetzung erst nach Erreichen einer hohen Modellkomplexität angestoßen werden kann. Danach wird die Anpassung fortgesetzt, indem wieder Terme hinzugefügt werden.

Das probeweise Hinzufügen von je zwei oder mehr Termen pro Schritt zwecks eines effektiveren Anpassungsschritts würde zu einem sehr hohen Rechenaufwand führen. Zum Beispiel würden bei 20 Eingangsparametern p = 1770 Terme bei einem Modell 3. Ordnung generiert werden, wodurch sich $p \cdot (p-1)/2 = 1.565.565$ mögliche Kombinationen von je zwei Termen im ersten Iterationsschritt ergeben würden.

Ein Iterationsverlauf wird durch die Anzahl δ^+ der in jedem Schritt hinzugefügten und der Anzahl δ^- der danach mindestens zu entfernenden Terme vorgegeben, z. B. $\delta^+ = 2$ Terme hinzufügen und $\delta^- = 1$ Term entfernen. Damit sich das Modell überhaupt entwickeln kann, muss offensichtlich $\delta^+ > \delta^-$ gelten. Die Wahl von $\delta^- = 1$ hat sich als zweckmäßig erwiesen, da ein Modell A mit δ_A^+ und $\delta_A^- = 1$ pro Schritt in etwa den gleichen Rechenaufwand benötigt wie ein Modell B mit $\delta_B^+ = \alpha \cdot \delta_A^+$ und $\delta_B^- = \alpha$ ($\alpha > 1$), Modell A aber öfter die Gelegenheit hat, ungünstigere Terme unmittelbar auszutauschen, so dass der nächste hinzugefügte Term basierend auf einem besseren Modell ermittelt wird.

In den ersten Iterationsschritten passt sich das Modell noch grob an und es findet keine Überanpassung statt, so dass selten mehr als δ^- Terme entfernt werden. Pro Schritt entstehen somit $\delta^+ - \delta^-$ neue Terme. Irgendwann ist die Anpassungsfähigkeit des Polynommodells an die Zielfunktion fast erreicht und der Beitrag der Terme, durch die das Modell zur Überanpassung tendieren kann, nimmt zu. In dieser Phase generiert das Modell in jeder Iterationsschleife im Mittel weniger als $\delta^+ - \delta^-$ Terme und optimiert die Termstruktur weiter, wobei der Fehler langsamer abnimmt als in der ersten Phase. In der letzten Phase hat sich das Modell dann so weit entwickelt, dass über mehrere Iterationsschritte keine nennenswerte Verbesserung mehr erzielt werden kann und die Anzahl der Terme um einen Mittelwert schwankt. Die Iteration kann dann beendet werden.

3.6 Numerische Umsetzung

Zur Berechnung der Least Squares-Lösungen der Gleichungssysteme 7.1.4 wurde eine QR-Zerlegung (Abschn. 3.1.1.4) mit einem Gram-Schmidt-Orthogonalisierungsverfahren gewählt [109]. Dabei werden die Spalten der Matrix A als Vektoren aufgefasst und beginnend mit der ersten Spalte nacheinander bezüglich aller vorhergehenden Spalten orthgonalisiert. Die so erhaltenen Vektoren u werden normiert und zur orthonormalen Matrix

$$Q = \left(\frac{u_1}{\|u_1\|} \mid \frac{u_2}{\|u_2\|} \mid \ldots \mid \frac{u_p}{\|u_p\|} \right) \tag{3.6.1}$$

zusammengefasst. Die obere Dreiecksmatrix R ergibt sich zu $R = Q^T A$. Das klassische Gram-Schmidt-Verfahren ist numerisch instabil. Es existieren aber Modifikationen zur Stabilisierung des Verfahrens [109].

Wenn in den Erweiterungsschritten der adaptiven Anpassung alle noch nicht verwendeten Koeffizien-

ten nacheinander geprüft werden sollen, muss die QR-Zerlegung nicht jedes mal komplett neu berechnet werden. Sei A die aktuelle Modellmatrix und z_i der Spaltenvektor, der die aus den Eingangsdaten berechneten Kandidatenterme enthält. Dann kann die QR-Zerlegung der um z_i erweiterten Matrix $A^+ = (A \mid z_i)$ aus der QR-Zerlegung von A berechnet werden, indem z_i bezüglich der $u_{1\ldots p}$ orthonormalisiert wird. Analog kann auch in den Vereinfachungsschritten die Matrix A^-, die aus A durch probeweises Auslassen je einer Spalte entsteht, aus der QR-Zerlegung von A berechnet werden, da das Entfernen eines orthonormalen Vektors die Orthogonalität der restlichen Vektoren nicht beeinflusst.

3.7 Rechenaufwand

Eine Erhöhung von k erhöht neben der Anzahl der Rechenschritte pro Iterationsschritt auch den Rechenaufwand der QR-Zerlegung: $O(2p^2 n_{train})$ für die Berechnung mit dem modifizierten Gram-Schmidt-Verfahren [109]. Hierbei ist p die Anzahl der aus den m Eingangsparametern gebildeten Terme im Modellpolynom. Für die Anzahl der Trainingsdaten gilt bei der k-fach Kreuzvalidierung $n_{train} = n\frac{k-1}{k}$.

3.8 Permutieren der Datensätze

Wenn in einem Schritt derjenige Term eliminiert werden soll, der im vorherigen Schritt hinzugefügt wurde, würde das Modell in eine Endlosschleife laufen. Dieser Fall tritt ein, wenn das Modell so weit entwickelt ist, dass keine weitere Verbesserung möglich ist. Er kann aber auch eintreten, wenn sich das Modell in diesem Iterationsschritt nicht ausreichend weit entwickeln konnte, um einen bestimmten Zusammenhang präziser wiederzugeben. Daher eignet sich dieser Fall nicht notwendigerweise als Kriterium für den Abbruch der Iteration.

Durch die zufällige Verteilung der Daten auf die Trainings- und Testdatenmenge (Gl. 1.6.5) soll gewährleistet werden, dass beide Datensätze ähnlich viel Information über den zugrundeliegenden Zusammenhang zwischen Eingangsgrößen und Zielgrößen erhalten. Wird ein Anpassungsschritt auf zwei unterschiedlichen zufälligen Verteilungen gemacht, dann ist die Anpassungsgüte gleicher Kandidatenterme auf den unterschiedlichen Modellen im Allgemeinen nicht gleich. Der Unterschied zwischen den beiden Modellen wird umso größer, je stärker das Rauschen ist und je ungleichmäßiger die Information verteilt ist. Das Modell sieht demnach bei einer zufälligen Umverteilung der Daten auf Trainings- und Testmenge eine unterschiedliche Fehlerlandschaft. Der Unterschied ist unter der Annahme gleichmäßig verteilter Information aber nur gering, so dass gute Terme immer für relativ kleine Fehler und schlechte Terme für relativ große Fehler sorgen.

Durch eine Umverteilung ist es somit möglich, dass sich die nach Anpassungsgüte sortierte Reihen-

folge der Kandidatenterme geringfügig ändert und der beste Term ein anderer ist als bei dem vorherigen Modell. Dadurch kann die Endlosschleife vermieden werden und die Iteration kann fortgesetzt werden. Alternativ zu einer Umverteilung könnte auch z. B. der zweitbeste Term hinzugefügt werden. Das Permutieren der Daten hat aber den Vorteil, dass sich das Modell weniger auf spezielle Eigenschaften einer gegebenen Verteilung von Test- und Trainingsdaten anpasst.

Die Iteration wird ebenfalls in eine Endlosschleife laufen, wenn durch Entfernen von Termen wieder ein Modell entsteht, das bereits in einem vorherigen Iterationsschritt durchlaufen wurde. Dies kann ebenfalls durch eine Permutation der Daten vermieden werden. Eine praktikable Überprüfung auf eine solche Schleife basiert auf der dazu notwendigen Bedingung, dass der Fehler des momentanen Modells gleich dem eines älteren Modells mit gleicher Anzahl Terme ist. Dazu wird in jedem Durchlauf der momentane Fehler und die Anzahl der Terme gespeichert und mit allen bisherigen Iterationsschritten zu gegebener Anzahl Terme verglichen. Falls durch dieses nicht hinreichende Kriterium öfter als nötig permutiert wird, wird die Modellanpassung dadurch nicht gestört, da die Annahme gleichmäßig verteilter Daten auf die Trainingsdatensätze besteht. Sie kann lediglich verlangsamt werden, da durch die Datenpermutation immer wieder in geringfügig unterschiedliche Richtungen optimiert wird. Dies kann die Zwischenschritte der Rekonstruktion eines Zusammenhangs stören. Eine Permutation nach jedem Erweiterungsschritt würde die weitere Entwicklung allerdings zu stark behindern.

Beim Trainieren auf unterschiedlichen Datensätzen werden ähnliche Modelle konstruiert, die sich auf die jeweiligen Trainingsdatensätze spezialisieren. Einer Überanpassung wird dabei durch gelegentliches Permutieren der Daten entgegengewirkt. Somit können sie die allen Datensätzen gemeinsame Information besser approximieren. Die Iteration kann abgebrochen werden, wenn sich der Testfehler über mehrere Permutationszyklen nicht mehr wesentlich ändert.

Eine Permutation der Daten führt wegen der Tendenz zur Spezialisierung auf das jeweilige Ensemble aus Testdatensätzen oft zu einer Erhöhung des Testfehlers auf den permutierten Daten, während sich der Trainingsfehler dabei auch verringern kann. Große Sprünge im Testfehler bei einer Neuverteilung der Daten sind ein Hinweis darauf, dass der Informationsgehalt der Daten relativ klein im Verhältnis zur Anzahl der Eingangsgrößen ist oder dass die einzelnen Testdatensätze während der Kreuzvalidierung nicht immer repräsentativ verteilt sind. Die weniger repräsentativen Testdatensätze verschlechtern den über alle Testdatensätze gemittelten Fehler dann verhältnismäßig stark, da der quadratische Fehler als Maß gewählt wurde. Bei einer hohen Anzahl an Kreuzvalidierungsschritten sind die Testmengen klein und der Einfluss solcher Ausreißer kann umso größer werden.

3.9 Maximale Ordnung der Auswahlterme

Je höher die maximale Ordnung der Auswahlterme des Modells ist, desto komplexere Zielfunktionen können angepasst werden. Einer Überanpassung wird zwar durch Optimierung auf den Testfehler und Vereinfachung der Termstruktur entgegengewirkt, eine maximale Ordnung und damit hohe Anzahl an Auswahltermen erschwert aber die Optimierung. Die Fehlerlandschaft, die jedem Polynom einen Least Squares-Anpassungsfehler zuordnet, zerklüftet mit steigender Anzahl an Modelltermen immer stärker, d. h. es entstehen immer mehr lokale Minima. Daher besteht im Fall zu hoher maximaler Modellkomplexität die Gefahr, dass das Modell während der Erweiterung der Termstruktur in ein ungünstiges Nebenminimum konvergiert und schlechter abschneidet als eine Anpassung mit weniger Termen von niedrigerer maximaler Ordnung. Um dieses Risiko bei hoher geforderter Modellkomplexität dennoch einzuschränken, können mehrere Anpassungen unabhängig voneinander auf unterschiedlich permutierten Datensätzen gemacht werden, von denen dann die beste beibehalten wird.

Am Oszillationsmarken-Datensatz (Abschnitt 2.1.3: $n = 695$ Messungen, bei denen sich die Zielgröße vermutlich im Wesentlichen durch $m = 4$ Eingangsgrößen beschreiben lässt), wurden verschiedene maximale Ordnungen q_{max} und Kreuzvalidierungsstrategien verglichen. Die Anpassungsgüte R^2 der erreichten adaptiven Anpassung ist vergleichbar mit der einer empirisch ermittelten Anpassungsformel [21].

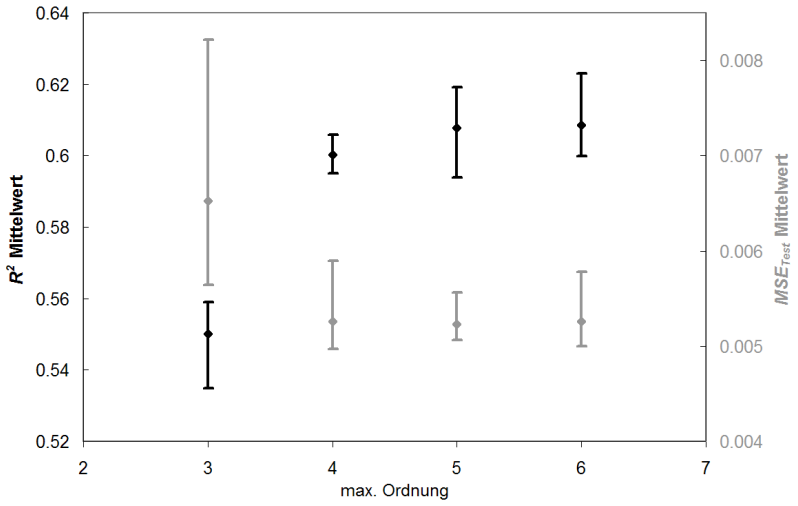


Abbildung 3.4: *Bestimmtheitsmaß R^2 und Testfehler MSE_{test} für Anpassungen verschiedener maximaler Ordnungen an einen Datensatz aus Oszillationsmarkenmessungen.*

Abbildung 3.4 zeigt das Bestimmtheitsmaß R^2 und den Testfehler MSE_{test} für Anpassungen mit höchsten Termordnungen $q_{max} = 3$ ($p = 34$ Kombinationen) bis $q_{max} = 6$ ($p = 209$ Kombinationen). Die Iterationen wurden gestoppt, nachdem sich Testfehler und Anzahl verwendeter Terme über eine

große Anzahl Iterationsschritte im Mittel nicht mehr änderten. Es ist der Mittelwert aus jeweils 20 Modelldurchläufen aufgetragen, wobei die Fehlerbalken Minimum und Maximum der berechneten Werte angeben.

Eine Auswahl aus Termen bis zur maximalen Ordnung 3 kann die Zielgröße weder bezogen auf das Bestimmtheitsmaß, noch auf den Testfehler so gut wiedergeben wie die Anpassungen höherer Ordnung. Im Mittel sind die Anpassungen 5. und 6. Ordnung nur geringfügig besser als die 4. Ordnung. Die mittlere Anzahl der verwendeten Terme beträgt im Modell 3. Ordnung 15,65 Terme, bei 4. Ordnung 18,65, bei 5. Ordnung 23,25 und bei 6. Ordnung 22,6 Terme. Die Modellpolynome enthalten dabei überwiegend Terme der höchsten kombinierten Ordnung, was auf eine hohe Wechselwirkung der Eingangsgrößen hinweist.

3.10 Abbruchkriterium

Wenn sich der Fehler durch Hinzufügen von Termen verschlechtert, wird die Komplexität des Modells in den folgenden Vereinfachungsschritten reduziert, solange sich der Fehler dadurch wieder verbessert. Dadurch wird der Fehler im Mittel gegen ein Minimum konvergieren bis auf Schwankungen durch Datenpermutation. Als robuste Fehlergröße der Iterationsschritte kann der gleitende Durchschitt der $MSE_{test}^{(1,\ldots,i)}$ in iteration i über eine vorgegebene Zahl von Iterationen g herangezogen werden:

$$MSE_{gleit}^{(i)} = \frac{1}{g} \sum_{j=i-g+1}^{g} MSE_{test}^{(i)} \tag{3.10.1}$$

Die Iteration zur Modellanpassung kann abgebrochen werden, wenn

- $MSE_{gleit}^{(i)}$ kleiner ist als eine vorgegebene Schranke. Diese kann absolut sein oder relativ zum Fehler zu Beginn der Iteration.
- $MSE_{gleit}^{(i-1)} / MSE_{gleit}^{(i)}$ sich während einer vorgegebenen Anzahl der letzten Iterationsschritte nicht stärker verändert als eine vorgegebene Schranke.
- eine vorgegebene maximale Anzahl an Iterationsschritten erreicht wurde.

Als finales Modell wird dasjenige aus den N Iterationsschritten gewählt, das den geringsten mittleren Vorhersagefehler erzielte:

$$M = \underset{1 \leq i \leq N}{\arg\min} MSE\left(M^{(i)}\right) \tag{3.10.2}$$

3.11 Normierung Transformation der Eingangsdaten

Die Anpassung eines vollständigen Polynoms n-ten Grades, d. h. dass alle Koeffizienten und Produkte daraus bis zum Grad n enthalten sind,

$$x_1^{a_1} \cdot x_2^{a_2} \cdot \ldots \cdot x_p^{a_p}, \ \sum_{j=1}^{p} a_p \leq n \tag{3.11.1}$$

ist invariant gegenüber Normierungen und Transformationen $x \to x'$ mit $x' = \alpha x - x_0$:

$$\begin{aligned} f(x) &= konst + a_1 x + a_2 x^2 + \ldots + a_n x^n \\ x &\to \alpha x - x_0 \\ \Longleftrightarrow f(x) &= \underbrace{konst - x_0 a_1 + x_0^2 a_2 + \ldots}_{konst'} + \underbrace{(\alpha a_1 - 2\alpha x_0 a_2 + \ldots)}_{a_1'} x' \\ &\quad + \underbrace{(\alpha^2 a_2 + \ldots)}_{a_2'} x'^2 + \ldots + a_n x'^n \end{aligned} \tag{3.11.2}$$

Dies gilt analog für vollständige Polynome mit mehreren Unbekannten, jedoch nicht mehr für die unvollständigen Polynome im Verlauf der Iteration. Die Entwicklung der Anpassung hängt daher von der Transformation der Eingangsparameter ab. Eine reine Normierung der Daten gemäß $x \to x'$ mit $x' = \alpha x$ hat dagegen keinen Einfluss auf die Anpassungsentwicklung, sondern führt lediglich zu einer geänderten Skalierung der Regressionskoeffizienten.

Als Beispiel wird das vollständige Polynom $f(x) = (x-1)^2 + \varepsilon$ mit dem zufälligen Fehler ε durch die Transformation $x' = x - 1$ auf das unvollständige Polynom $f(x') = x'^2 + \varepsilon$ abgebildet. Die Adaption an Funktionswerte der untransformierten Funktion $f(x)$ würde daher in mehreren Schritten erfolgen und zu einem Polynom mit zwei Termen anstatt einem führen. Die Konstante sei hierbei nicht mitgezählt. Eine höhere Anzahl an Termen erhöht aber die Gefahr der Überanpassung.

Wenn kein Vorwissen bezüglich einer zweckmäßigen Transformation der Eingangsdaten x_i, $1 \leq i \leq m$ existiert, sollte die Entwicklung des Modells idealerweise um einen Punkt $w = (w_1, \ldots, w_m)$ stattfinden, in dessen Umgebung sich möglichst viele Daten befinden. Dadurch kann sich das Modell der Zielfunktion in diesem gut mit Daten belegten Bereich ohne Umweg über unnötig viele Kompromissterme (siehe Abschnitt 3.5) anpassen. Bei gleichmäßiger Verteilung der Daten über den gesamten Datenraum kann hierzu einfach die Intervallmitte $w_i = \frac{1}{2}\left(\max_j (x_i)_j - \min_j (x_i)_j\right)$, $1 \leq j \leq n$ gewählt werden.

Wenn die Daten aber stark clustern, ist die optimale Wahl eines Entwicklungsursprungs aber nicht

einfach und nicht notwendigerweise eindeutig. Wird für jedes i willkürlich ein Punkt w_i gewählt, der lediglich innerhalb irgendeines Clusters aus der Menge der i-ten Komponenten der Daten $\left\{(x_{1\ldots m})_j\right\}$ liegt, dann kann es im mehrdimensionalen Fall vorkommen, dass w in einem Gebiet liegt, das nicht mit Messdaten abgedeckt ist.

Der Mittelwert der Komponenten $w_i = \frac{1}{n}\sum_{j=1}^{n}(x_i)_j$ eignet sich als Entwicklungspunkt, da er die Verteilung aller Cluster gewichtet.

3.12 Vergleiche der Fehlerabschätzung

3.12.1 Überanpassung bei AIC

Es wurden 30 Wertepaare $(x_i, f(x_i))$ generiert mit $f(x) = x^2 + \Delta$, $-1 \leq x \leq 1$ und einer aus dem Intervall $[-0.5; 0.5]$ gleichverteilten Zufallszahl Δ. Die Daten wurden mit einer Stepwise Forward Selection mit den Auswahltermen $\left\{x^i \mid i = 1, ..., 6\right\}$ angepasst und der Verlauf von AIC und BIC (Abschnitt 35 und 35) verfolgt. AIC nahm entlang des Iterationspfads $\left\{M^{(1)}, ..., M^{(6)}\right\}$ monoton ab, so dass die Anpassungsfunktion mit minimalem AIC alle Terme enthielt und damit eine deutlich zu hohe Komplexität annahm. BIC dagegen erzielte wegen der stärkeren Bestrafung erhöhter Modellkomplexität bei $M^{(1)} = konst^{(1)} + a_1^{(1)} \cdot x^2$ bereits das Minimum, so dass eine optimale Anpassung gefunden wurde.

Der hier vorgestellte adaptive Algorithmus wählte dagegen mit 4-fach Kreuzvalidierung das Modell $M = konst + a_1 \cdot x + a_2 \cdot x^2$.

3.12.2 Unteranpassung bei BIC und verfrühtes Abbrechen

Um Probleme bei der Unteranpassung und das verfrühte Abbrechen bei Betrachtung der Fehlerveränderung einer Forward Selection zu veranschaulichen, seien die Funktionen

$$\begin{gathered} f(x) = \tanh(x+3\pi) - \tanh(x+\pi) + \tfrac{1}{2}(\tanh(x-\pi)\tanh(x-3\pi)) \\ f_\Delta(x) = f(x) + \Delta \\ -4\pi \leq x \leq 4\pi, \Delta = \text{gleichverteilte Zufallszahl } \in [-0.5; 0.5] \end{gathered} \tag{3.12.1}$$

gegeben, die zwei lokale Maxima aufweist (Abbildung. 3.5). $f(x)$ könnte z. B. das Ein- und Ausschalten zweier um $x = \pm\pi$ zentrierter und voneinander separierter Effekte darstellen, $f_\Delta(x)$ eine verrauschte Messung davon. Mit dieser Funktion wurden 40 Datenpunkte an zufällig gewählten Stützstellen generiert.

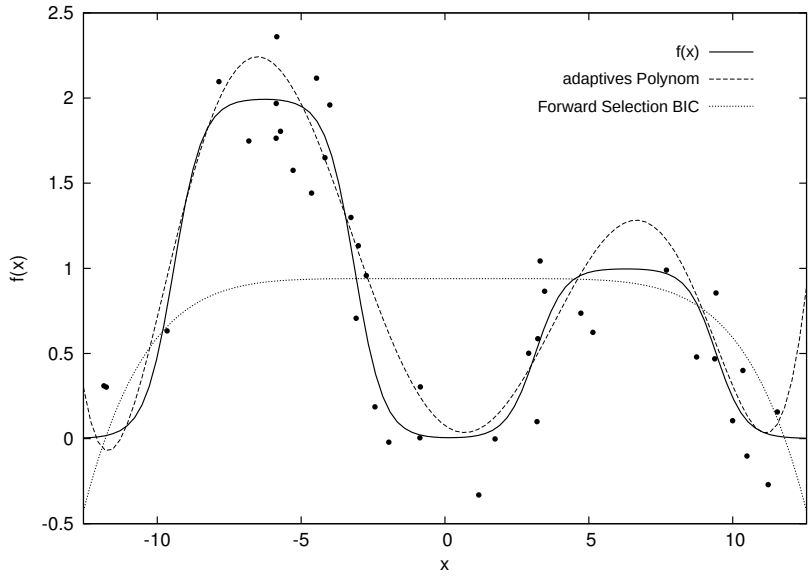


Abbildung 3.5: *f(x), die mit $f_\Delta(x)$ generierte Punkteschar und die Fit-Funktionen der Forward Stepwise Regression mit BIC sowie des adaptiven Algorithmus aus dieser Arbeit.*

Zur Anpassung an die Daten wurde eine Stepwise Forward Selection verwendet, die abgebrochen wurde, sobald sich das Akaike-Informationskriterium bzw. Bayes-Informationskriterium bezüglich des Fehlers auf den Lerndaten nicht mehr weiter verbesserte.

Dem Regressionsverfahren wurden die Auswahlterme $\left\{x^i \mid i = 1, ..., 6\right\}$ zur Verfügung gestellt. Im ersten Iterationsschritt erzielte das Modell $M^{(1)} = konst^{(1)} + a_1^{(1)} \cdot x^6$ die beste Anpassung. Im zweiten Schritt nahm BIC wieder zu und die Iteration wurde abgebrochen. Das gewählte Modell $M^{(1)}$ stellt aber nur ein lokales Optimum entlang des Iterationspfads $\left\{M^{(1)}, ..., M^{(6)}\right\}$ dar. Ab dem 3. Iterationsschritt hätte BIC wieder deutlich abgenommen, da hier erst eine ausreichend hohe Modellkomplexität erreicht wurde, um die beiden Maxima zumindest grob voneinander trennen zu können. Abbildung 3.6 zeigt den Verlauf von AIC, BIC und des mittleren quadratischen Fehlers MSE_{valid} auf einem durch $f_\Delta(x)$ gebildeten separatem Validierungsdatensatz mit 40 Datenpunkten.

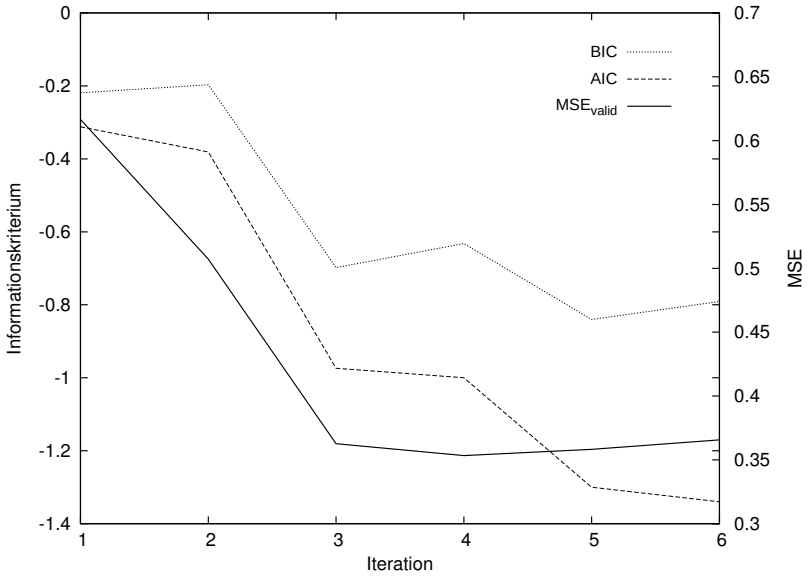


Abbildung 3.6: *Verlauf von AIC und BIC während des Iterationspfads der Forward Selection.*

Im Vergleich dazu wählte der hier vorgestellte adaptive Algorithmus mit einer 4-fach Kreuzvalidierung in 30 Testläufen fast immer alle sechs Terme aus, obwohl der MSE_{test} in manchen Iterationsschritten zunahm. Durch spätere Ersetzungen redundanter Terme wurde die Anpassungsfähigkeit des finalen Modells verbessert. Abbildung 3.5 vergleicht die die Ergebnisse mit der zugrundeliegenden Funktion $f(x)$.

3.13 Modellierung einfacher Polynome

3.13.1 Vorbetrachtung

Aus Polynomfunktionen generierte Datenmengen $\{(x_i, y_i)\}$ mit Stützstellen x_i werden zur Überprüfung des adaptiven Modells auf die Fähigkeit, das zugrundeliegende Polynom rekonstruieren zu können, verwendet.

Zunächst werden nicht verrauschte eindimensionale Daten betrachtet, die aus einer beliebigen Vorgabepotenz p_v gebildet werden, $y(x) = a \cdot x^{p_v}$. Das adaptive Polynom wertet die Daten mit einem maximalen Polynomgrad $p \geq p_v$ aus. Für die Aufteilung der Daten auf Trainings- und Testdaten wird $n_{train} > p_v + 1$ und $n_{test} > p_v + 1$ gewählt.

Für den mittleren quadratischen Fehler $MSE = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} \left\| \alpha_{LS} \cdot x_i^p - a \cdot x_i^{p_v} \right\|^2$ gilt: $MSE = 0$ für $p = p_v$ und $\alpha_{LS} = a$. α_{LS} ist der Koeffizient der Least Squares-Anpassung. Weiterhin ist $MSE > 0$ für alle nicht passenden Potenzen $p \neq p_v$. Damit wählt das Modell bereits im ersten Schritt die passende Potenz.

Wird den Messdaten ein zufälliger gleichverteilter Fehler ε_i mit Mittelwert 0 beaufschlagt, $y_\varepsilon(x_i) = a \cdot x_i^{p_v} + \varepsilon_i$, dann ist

$$MSE = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} \left\| \alpha_{LS} \cdot x_i^p - \left(a \cdot x_i^{p_v} + \varepsilon_i \right) \right\|^2$$

ebenfalls minimal für $p = p_v$ und $\alpha_{LS} = a$. Bei einer Least Squares-Abschätzung $\alpha_{LS} \approx a$ wählt das Modell daher als erste Potenz x^{p_v}, wenn das Rauschen ε_i im Verähltnis zu n den funktionalen Zusammenhang $y_\varepsilon(x)$ nicht zu stark überdeckt.

3.13.2 Beispielfälle

In den folgenden Beispielen wurden eindimensionale Daten aus einem einzelnen Potenzterm $y(x_i) = x_i^z$, $z = 0, 1, 2, 3$ gebildet und mit einer Genauigkeit von neun Nachkommastellen gespeichert. Die Stützstellen wurden gemäß $x_i \in [0; 1]$, $i = 1, ..., 100$ zufällig gewählt. Zusätzlich wurden weitere Datenmengen generiert, indem den Funktionswerten ein zufälliges Rauschen ε mit unterschiedlichen Intensitäten überlagert wurde.

Die Anpassung der Modelle erfolgte mit einer maximalen Ordnung der Auswahlterme $q_{max} = 5$ und einem Entwicklungsschema von schrittweise $\delta^+ = 2$ hinzugefügten und $\delta^- = 1$ entfernten Auswahltermen. Die Daten wurden zufällig zu 75% auf Trainings- und zu 25% auf Testdaten aufgeteilt. Der für die Modellentwicklung verwendete MSE_{test} wurde über eine 4-fach Kreuzvalidierung gemittelt.

Konstanter Term

Die Konstante wurde in allen Datensätzen

$$y_\varepsilon(x_i) = 1 + \varepsilon_i \tag{3.13.1}$$

erkannt und ergab sich zu:

$konst(\varepsilon_i = 0) = 1$,
$konst(\varepsilon_i \in [\pm 0,05]) = 1,003$,
$konst(\varepsilon_i \in [\pm 0,1]) = 1,010$,
$konst(\varepsilon_i \in [\pm 0,25]) = 1,018$,
$konst(\varepsilon_i \in [\pm 0,5]) = 1,068$ und
$konst(\varepsilon_i \in [\pm 1]) = 1,225$.

Zusätzlich enthielten die Modelle einige Terme aus der Menge der restlichen Auswahlterme $\{x, x^2, x^3, x^4, x^5\}$, deren Koeffizienten sich mit steigendem Rauschen betragsmäßig erhöhten.

Linearer Term

Um die Anpassung an die linearen Funktion zu bewerten, wurden die Bestimmtheitsmaße

R^2_{real} zwischen den Wertepaaren $y(x_i) = x_i$ und

$$y_\varepsilon(x_i) = x_i + \varepsilon_i \tag{3.13.2}$$

sowie

- R^2_{modell} zwischen den Wertepaaren $y(x_i) = x_i$ und den Modellwerten

gegenübergestellt. Weiterhin ist die Anzahl der zuzätzlich aus den fünf übrigen Auswahltermen ins Modell aufgenommenenn Terme in Tabelle 3.1 aufgelistet.

Potenzterm x + Rauschen $\varepsilon \in$	R^2_{real}	R^2_{modell}	Term erkannt	zusätzliche Terme
0	1	1	ja	4
$[\pm 0,05]$	0,9917	0,9918	ja	2
$[\pm 0,1]$	0,9637	0,9640	ja	2
$[\pm 0,25]$	0,7994	0,7999	ja	3
$[\pm 0,5]$	0,4017	0,4234	ja	2
$[\pm 1]$	0,0087	0,1947	ja	3

Tabelle 3.1: *Anpassung des adaptiven Polynommodells an eine lineare Funktion.*

Der lineare Term wurde in allen Fällen erkannt. Die zusätzlichen, in der zugrundeliegenden Funktion $y(x_i)$ nicht enthaltenen Terme erhielten bei der Anpassung an die nicht verrauschten Daten Koeffizienten der Größenordnung 10^{-16}. Mit zunehmendem Rauschen wurde deren Betrag größer und bei $y_\varepsilon(x_i) = x_i + \varepsilon_i \in [\pm 0,25]$ erreichten einige bereits die Größenordnung 10^{-1}. Bei dieser Funktion schlägt sich der Informationsverlust durch das Rauschen deutlich in der geringeren Korrelation $R^2_{real} = 0,7994$ nieder. Bei $y_\varepsilon(x_i) = x_i + \varepsilon_i \in [\pm 1]$ ist die Korrelation fast nicht mehr erkennbar. Bis auf diesen Fall geben die Modelle die Daten mit einem vergleichbaren Korrelationskoeffizienten wieder wie die zugrundeliegende Funktion selbst. Dies ist ein Indiz dafür, dass die Modelle die Daten weder zu ungenau abbilden ($R^2_{modell} \ll R^2_{real}$), noch überanpassen ($R^2_{modell} \gg R^2_{real}$).

Quadratischer Term

Tabelle 3.2 zeigt die Anpassung an

$$y_\varepsilon(x_i) = x_i^2 + \varepsilon_i \tag{3.13.3}$$

Der quadratische Term wurde in allen Fällen erkannt. Die Anzahl der zusätzlichen Terme nahm mit stärker werdendem Rauschen zu. Die Korrelation der Modelldaten ist der Korrelation der Daten mit $y(x_i)$ sehr ähnlich.

Potenzterm x^2 + Rauschen $\varepsilon \in$	R^2_{real}	R^2_{modell}	Term erkannt	zusätzliche Terme
0	1	1	ja	2
$[\pm 0,05]$	0,9903	0,9904	ja	2
$[\pm 0,1]$	0,9590	0,9592	ja	2
$[\pm 0,25]$	0,7910	0,7911	ja	3
$[\pm 0,5]$	0,4782	0,4824	ja	4
$[\pm 1]$	0,1885	0,2030	ja	4

Tabelle 3.2: *Anpassung des adaptiven Polynommodells an eine quadratische Funktion.*

Kubischer Term

Auch im Fall

$$y_\varepsilon(x_i) = x_i^3 + \varepsilon_i \tag{3.13.4}$$

(Tabelle 3.3) ist die Modellkorrelation derjenigen von $y(x_i)$ ähnlich. Ab einem Rauschen $\varepsilon_i \in [\pm 0,5]$ wurde der Term x^3 allerdings nicht mehr erkannt, so dass die Modelle die Funktion nicht mehr korrekt abbilden konnten. In diesen Fällen legt die grafische Auftragung (Abb. 3.7) der Daten ebenso wie das niedrige R^2_{real} nahe, dass $y(x_i) = x_i^3$ unter dem starken Rauschen bereits nicht mehr genau erkennbar ist.

Potenzterm x^3 + Rauschen $\varepsilon \in$	R^2_{real}	R^2_{modell}	Term erkannt	zusätzliche Terme
0	1	1	ja	2
$[\pm 0,05]$	0,9895	0,9895	ja	2
$[\pm 0,1]$	0,9630	0,9638	ja	2
$[\pm 0,25]$	0,7698	0,7699	ja	2
$[\pm 0,5]$	0,4758	0,5020	nein	3
$[\pm 1]$	0,0715	0,0853	nein	3

Tabelle 3.3: *Anpassung des adaptiven Polynommodells an eine kubische Funktion.*

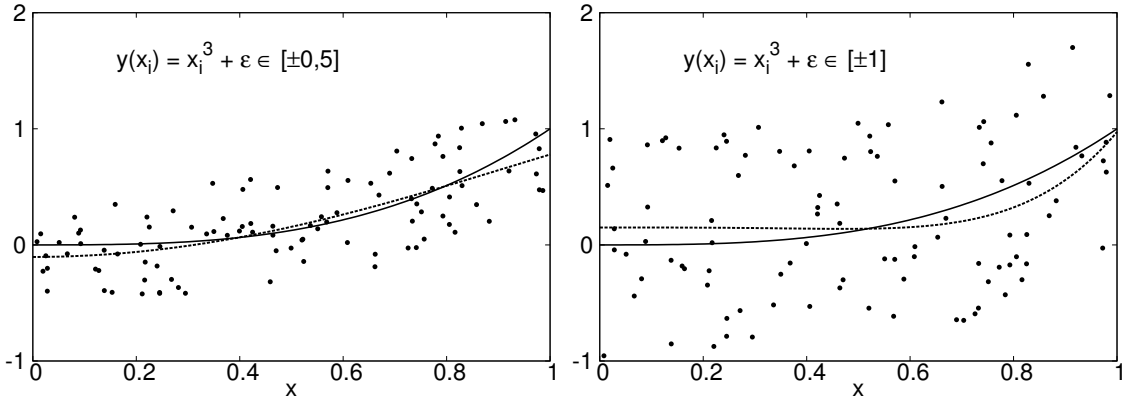


Abbildung 3.7: *Datenpunkte, zugrundeliegende Funktion (durchgezogene Linie) und adaptive Polynomfunktion (gestrichelte Linie) für die stark verrauschten kubischen Daten.*

4-dimensionales Polynom

Zur Bewertung der Anpassungsfähigkeit an ein mehrdimensionales Polynom wurde

$$\begin{gathered} y_{\varepsilon}^{4d} : \mathbb{R}^4 \to \mathbb{R}, \\ y_{\varepsilon}^{4d}\left((x_1, x_2, x_3, x_4)_i\right) = (x_1, x_2, x_3, x_4)_i + \left(x_1^2, x_2^2, x_3^2, x_4^2\right)_i + \left(x_1^3, x_2^3, x_3^3, x_4^3\right)_i + \varepsilon_i \end{gathered} \tag{3.13.5}$$

gewählt und wie in den vorherigen Beispielen mit $q_{max} = 5 \Rightarrow 126$ Auswahltermen ausgewertet. Die Iteration wurde abgebrochen, wenn sich der MSE_{Test} während 50 Erweiterungsschritten um nicht mehr als 1% geändert hat. Zunächst wurden 100 Datenpunkte zufällig generiert, was bei der Komplexität dieses Problems bereits einer relativ dünnen Besetzung entspricht. Tabelle 3.4 fasst die Ergebnisse der Anpassung zusammen. R^2_{modell} und R^2_{real} sind durchgehend ähnlich, allerdings beinhaltet das adaptive Polynommodell bereits ab einem Rauschen $\varepsilon_i \in [\pm 0,05]$ nicht mehr alle Terme von $y^{4d}(x_i)$.

$y^{4d}(x)$ + Rauschen $\varepsilon \in$	R^2_{real}	R^2_{modell}	erkannte Terme (aus 12)	zusätzliche Terme (aus 114)
0	1	1	12	2
$[\pm 0,01]$	1	1	12	2
$[\pm 0,05]$	0,9997	0,9998	10	7
$[\pm 0,1]$	0,9990	0,9994	8	20
$[\pm 0,25]$	0,9938	0,9962	5	17
$[\pm 0,5]$	0,9741	0,9805	8	8
$[\pm 1]$	0,8920	0,9376	3	18

Tabelle 3.4: *Anpassung des adaptiven Polynommodells an ein 4-dimensionales Polynom dritten Grades mit 100 Datenpunkten.*

Anschließend wurde der Datensatz auf 1000 Datenpunkte vergrößert (Tabelle 3.5), wodurch eine genauere Rekonstruktion möglich wurde. Hierbei erkannte das Modell erst ab $\varepsilon_i \in [\pm 0,25]$ nicht mehr alle Terme der zugrundeliegenden Funktion.

$y^{4d}(x)$ + Rauschen $\varepsilon \in$	R^2_{real}	R^2_{modell}	erkannte Terme (aus 12)	zusätzliche Terme (aus 114)
0	1	1	12	5
$[\pm 0,01]$	1	1	12	7
$[\pm 0,05]$	0,9997	0,9997	12	4
$[\pm 0,1]$	0,9990	0,9990	12	7
$[\pm 0,25]$	0,9938	0,9939	8	8
$[\pm 0,5]$	0,9748	0,9759	8	16
$[\pm 1]$	0,9059	0,9091	5	11

Tabelle 3.5: *Anpassung des adaptiven Polynommodells an ein 4-dimensionales Polynom dritten Grades mit 1000 Datenpunkten.*

3.14 Capped data

3.14.1 Motivation

Der Messbereich von Messgeräten ist aus technischen Gründen oft nach oben oder unten beschränkt. Dadurch kann es zu einer Beschneidung („Cap") von Messwerten kommen. In der Realität existiert aber auch eine Vielzahl von Fällen, bei denen eine Messgröße intrinsisch durch Schwellwerte beschnitten

wird, die eine direkte statistische Betrachtung der zugrundeliegenden Zusammenhänge erschwert. Beispiele hierfür sind:

- Eine Qualitätskontrolle ermittelt, ob ein Produkt fehlerhaft ist oder nicht. Wenn der auftretende Fehler anstatt einer rein binären Entscheidung Fehler/fehlerfrei zusätzlich quantifiziert werden kann, so dass höhere Fehlerwerte einem schlechteren Produkt entsprechen, dann enthält eine Messung „Fehler = 0" einen geringeren Informationsgehalt als eine Messung „Fehler > 0".

- Beim äußeren fotoelektrischen Effekt besteht ein linearer Zusammenhang zwischen der maximalen kinetischen Energie E der herausgelösten Elektronen und der Frequenz ν des Lichts: $E = h \cdot \nu - W$. W ist hierbei die Austrittsarbeit, die nötig ist, um die Elektronen aus dem Metall herauszulösen. Unterhalb dieser Energie werden keine Elektronen herausgelöst und folglich ist dort $E(\nu) \equiv 0$.

- Eine Intensitätsmessung mit einem Fotosensor ist durch den Messbereich nach oben und unten beschränkt. Zu helle Signale übersteuern und sind nicht von dem maximalen messbaren Wert unterscheidbar. Analog dazu sind zu dunkle Signale nicht mehr von einer Signalstärke Null unterscheidbar.

- Die Kühlwassertemperatur T eines Motors mit offenem Külsystem (z. B. mit Überdruckventil und Ausgleichsbehälter) ist von der Motorleistung P abhängig. Bei gegebener Außentemperatur stellt sich zu jeder gegebenen Leistung eine bestimmte Temperatur $T(P)$ ein. Sobald das Kühlwasser siedet, erhöht sich dessen Temperatur nicht weiter und die Funktion $T(P)$ ist durch den Siedepunkt gedeckelt.

- Bei Bruchversuchen von Bauteilen kann die zum Bruch erforderliche Belastung als Gütemaß verwendet werden. Bauteile, deren Festigkeit höher ist als die maximale durch die Messvorrichtung erreichbare Belastung, können auf diese Weise nicht direkt miteinander bezüglich ihrer maximalen Belastungsfähigkeit verglichen werden.

- Im Beispiel HIC (Abschnitt 2.1.2) ist der *CAR* dadurch beschränkt, dass es keine Risse mit negativer Fläche geben kann.

Hiervon sind die Fälle zu unterscheiden, in denen eine Messgröße gegen einen konstanten Wert konvergiert, z. B. die Anzahl N der Objekte bei einem exponentiellen zeitlichen Zerfall: $N \to 0$ für $t \to \infty$.

3.14.2 Anforderung

Es soll eine Anpassung an schwellwertbehaftete Messwerte $Mess_{1,\ldots,n} \geq 0$ durch eine stetig differenzierbare Funktion Fit bezüglich einer Fehlernorm $\sum_{i=1}^{n} \|Mess_i - Fit_i\|$ an den Stützstellen i gemacht werden.

Beliebige obere oder untere Schwellwerte können analog implementiert oder durch entsprechende Transformation der Eingangsdaten auf einen Schwellwert 0 zurückgeführt werden.

Dabei können die Messwerte im Parameterraum in Gebiete mit zwei Eigenschaften aufgeteilt werden. Die einen enthalten konstante Messwerte zum Schwellwert $Mess_i = 0$, d. h. dass bei Variation eines Eingangsparameters keine Variation der Zielgröße stattfindet. Die anderen enthalten die variablen Messwerte $Mess_j > 0$. Die Übergänge zwischen diesen Gebieten schränken die Anpassung durch glatte Funtionen wie Polynome zusätzlich ein, da die Approximation solcher nicht differenzierbarer Stellen eine hohe Modellkomplexität erfordert. Mit Erhöhung des Anpassungsgrades steigt aber das Risiko einer Überanpassung. Die Modellierung der Übergänge erfordert eine erhöhte Anpassungsfähigkeit des Modells. Dadurch wird dem Modell Flexibilität zur Anpassung an die Gebiete mit positiven Messwerten genommen.

Um die Probleme beim Übergang zu umgehen, könnte die Anpassung lediglich auf Datensätze mit positiven Messwerten reduziert werden. Die Daten aus der Teilmenge mit $Mess_i = 0$ sind untereinander nicht vergleichbar und enthalten daher weniger Information als die positiven Werte. Sie enthalten allerdings die Information, dass in diesem Gebiet keine positiven Werte erlaubt sind. Durch eine Unterschlagung dieser Teilmenge würde das Modell nicht daran gehindert, in diesen Bereichen zugunsten der Anpassung an die restlichen Messpunkte beliebig hohe Werte zu generieren und somit unbrauchbare Vorhersagen zu machen. Eine Ausdünnung des Datensatzes ist bei hochdimensionalen Problemen ohnehin unerwünscht. Abbildung 3.8 zeigt ein Beispiel, in dem zwei Polynomanpassungen an Messdaten mit Capped Data-Eigenschaft verglichen werden. Bei der Anpassung im Fall a) wurden alle Werte $Mess_i = 0$ ignoriert, im Fall b) wurden sie berücksichtigt.

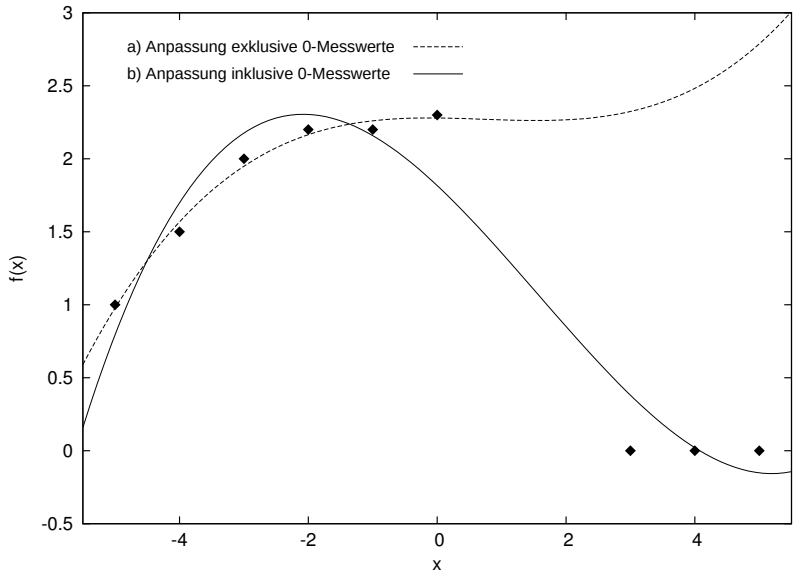


Abbildung 3.8: *Zwei Anpassungen eines Polynoms 3. Grades an exemplarische Messpunkte mit Capped Data-Eigenschaft. Im Fall a) wurden alle Messerte = 0 unterschlagen, im Fall b) erfolgte die Anpassung an alle Messpunkte. Fall a) ergibt zwar eine bessere Anpassung an die Messwerte > 0, die Vorhersage für die Messwerte an den Stützstellen x > 3 ist aber unbrauchbar.*

Eine Extrapolation aus Bereichen mit positiven Messwerten in $Mess_i = 0$-Bereiche kann zu negativen Fit - Werten führen. Solche Werte gilt es im Anwendungsfall zu interpretieren.

Da in der Literatur keine Verfahren bekannt sind, die Eigenschaften zweier Klassen $Mess_i = 0$ und $Mess_j > 0$ bei der Polynomanpassung explizit zu berücksichtigen, wird im Folgenden ein neuer eigener Anpassungsalgorithmus vorgestellt.

Ein denkbarer Ansatz wäre, die Koeffizienten anstatt durch eine least-squares Anpassung iterativ zu bestimmen, z. B. mittels eines Gradientenabstiegsverfahrens. Dabei könnte der Anpassungsfehler für alle $Mess_i = 0$-Werte Null definiert werden, wenn der dazugehörige Fit-Wert ≤ 0 ist. Dadurch würden die $Mess_i = 0$-Werte das Polynom nicht zu Null-Werten hin verzerren und folglich bliebe mehr Anpassungsflexibilität für die $Mess_j > 0$-Werte, ohne dass der Grad des Polynoms erhöht werden müsste. Bei einer iterativen Koeffizientenanpassung besteht aber die Gefahr, lediglich ein lokales Optimum zu finden. Daher soll der Least Squares-Ansatz, der ein globales Optimum ermittelt, beibehalten werden.

3.14.3 Beispiel HIC

Zur Vermeidung des HIC-Problems sind gerade die Bereiche mit $CAR = 0$ interessant. Ein Modell, das nur mit $CAR > 0$-Werten trainiert wurde, muss in diese Bereiche extrapolieren. Extrapolationen sind aber insbesondere bei den stark verrauschten HIC-Daten störanfällig.

Durch die möglichst exakte Wiedergabe von $CAR = 0$-Messwerten würde lediglich eine Charakteristik der Messmethodik angepasst anstatt einer physikalischen Kenngröße, welche die Unterscheidung der HIC-Resistenz zweier verschiedener $CAR = 0$-Proben ermöglicht.

Die Physik der Rissentstehung wird durch

- die Anzahldichte der möglichen Rissentstehungszentren
- die Festigkeit des Materials gegenüber Rissausbreitung
- den Rissausbreitungswiderstand und die Anzahldichte der Zentren, an denen ein sich ausbreitender Riss wieder gestoppt werden kann

beschrieben. Erst wenn die hierdurch gegebene Barriere überwunden wird, kann ein Riss entstehen. Diese Barriere ist die Ursache für den Schwellwert 0 der Messung. Wird sie ausgehend von einem Herstellungskonzept mit $CAR = 0$ noch weiter erhöht, ergibt sich eine Probe mit noch höherer HIC-Resistenz und der dazugehörige Messwert wäre ebenfalls $CAR = 0$. Ein Modell, das die aus den CAR-Messwerten gewonnene Information zur HIC-Resistenz glatt in die Bereiche fortsetzt, die durch die Capped data-Eigenschaft mit $CAR = 0$ gemessen wurden, könnte dort negative Werte vorhersagen. Dann wird ein Vergleich der HIC-Resistenzen von Punkten in diesen Bereichen ermöglicht, indem Fit-Werte eine umso bessere HIC-Güte beschreiben, je kleiner sie sind. Da die negativen Werte eine nicht messbare Zusatzinformation darstellen, ist zum Vergleich der Modellwerte mit den Messwerten das Fehlermaß $\|CAR_i - \Theta(Fit_i) \cdot Fit_i\|$ mit der Heavyside-Funktion

$$\theta(x) = \begin{cases} 0: & x < 0 \\ 1: & x \geq 0 \end{cases}$$

zu wählen.

3.14.4 Cap-Algorithmus

Motiviert durch dieses Problem wird ein neues iteratives Verfahren vorgestellt, bei dem die 0-Messwerte schrittweise durch negative Modellwerte ersetzt werden. Ein negativer Wert der Fit-Funktion $f(x_i)$ wird bei der Fehlerbewertung wie ein 0-Wert behandelt:

$$MSE(f(x_i), y_i) = (y_i - \max(0; f(x_i)))^2, y_i \in D_{train} \tag{3.14.1}$$

Algorithmus 3.2 setzt dies um.

Algorithmus 3.2 *Pseudocode des Cap-Algorithmus.*

$D_{train}{}^{(1)} = D_{train}$

für $t = 1, ..., n_{cap}$
  fit $f(x)^{(t)}$ an $D_{train}{}^{(t)}$
  für alle $y^{(t)} \in D_{train}{}^{(t)}$

$$y^{(t+1)} = \begin{cases} f(x)^{(t)} & , \text{ falls } y^{(1)} = 0 \text{ und } f(x)^{(t)} < 0 \\ y^{(1)} & , \text{ sonst} \end{cases}$$

Hierbei werden nach jedem Anpassungsschritt alle Werte der Zielgröße, deren Messwerte = 0 sind, durch den *Fit*-Wert ersetzt, falls dieser negativ ist. Dadurch werden nach und nach mehr $Mess_i = 0$-Werte durch extrapolierte negative Werte ersetzt. Negative *Fit*-Werte zu positiven Messwerten werden dagegen nicht ersetzt. Somit wird im Sinne der Messung keine Information verändert und der Anpassungsfehler durch den Knick beim 0-Übergang nimmt schrittweise ab.

Die Iteration kann abgebrochen werden, sobald sich der Anpassungsfehler nicht weiter als eine vorgegebene Schranke verbessert. Es kann vorkommen, dass die durch negative Fit-Werte ersetzten 0-Messwerte weit in den negativen Bereich driften. Das stellt aber kein Problem für die Anpassung der Funktion $\Theta(Fit_i) \cdot Fit_i$ dar, da die verschobenen 0-Messwerte in jedem Schritt lediglich in Richtung der extrapolierten Vorhersagen des restlichen Datensatzes konvergieren und somit nicht durch ihre eigene Manipulation der Zielfunktion unkontrolliert abdriften können.

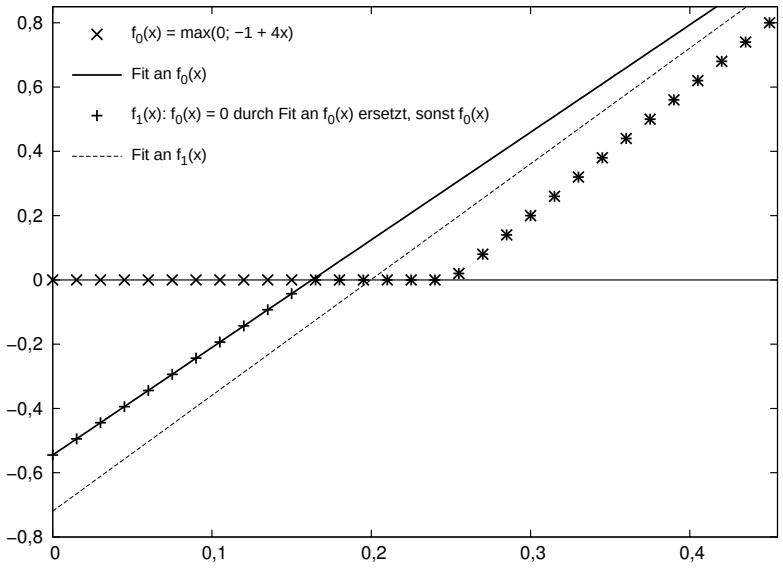


Abbildung 3.9: *Erster Iterationsschritt zur Anpassung an eine lineare Funktion mit Schwellwert 0.*

Abbildung 3.9 zeigt den ersten Iterationsschritt für eine lineare Funktion $f_0(x)$, bei der alle negativen

Funktionswerte Null gesetzt wurden. Die lineare Anpassung $Fit_{f_0(x)}$ approximiert die positiven Funktionswerte schlecht, da die Minimierung der Fehlerquadrate sie zu den Nullwerten hin abflacht. Im nächsten Schritt wird die neue Zielfunktion $f_1(x)$ aus $f_0(x)$ gebildet, indem die Werte $f_0(x) = 0$ durch negative *Fit*-Werte ersetzt werden und es wird eine Anpassung an $f_1(x)$ erstellt. Die Funktion $\Theta\left(Fit_{f_1(\mathrm{x})}\right) \cdot Fit_{f_1(\mathrm{x})}$ approximiert $f_0(x)$ besser als $\Theta\left(Fit_{f_0(\mathrm{x})}\right) \cdot Fit_{f_0(\mathrm{x})}$ und die Iterationsfolge $\Theta\left(Fit_{f_\mathrm{t}(\mathrm{x})}\right) \cdot Fit_{f_\mathrm{t}(\mathrm{x})}$ konvergiert gegen $f_0(x)$.

3.14.5 Konvergenz

Im Folgenden wird die Konvergenz des Cap-Verfahrens an eine durch den Schwellwert 0 nach unten beschränkte Ausgangsmenge von Messwerten $D = \left\{\left(x_i, y_i^{(1)}\right)\right\}$, $y_i^{(1)} \geq 0$ durch eine beliebige Anpassungsfunktion $f(x)$ gzeigt. Dazu werden die Messdaten in eine Menge $D_0^{(1)}$ mit und eine Menge $D_>^{(1)}$ ohne Schwellwert-Messwerte aufgeteilt:

$$\begin{aligned} D_0^{(1)} &:= \left\{\left(x_i, y_i^{(1)}\right) \in D \mid y_i^{(1)} = 0\right\} \\ D_>^{(1)} &:= \left\{\left(x_i, y_i^{(1)}\right) \in D \mid y_i^{(1)} > 0\right\} \end{aligned} \tag{3.14.2}$$

Ausgangsschritt (1):

$$f(x)^{(1)} := \left\| f(x)^{(1)} - y^{(1)} \right\|_2^2 = \min,\ (x, y) \in D \tag{3.14.3}$$

An den Stützstellen $\{x_i\}$ gilt:

$$\sum_{(x_i, y_i) \in D} \left(f(x_i)^{(1)} - y_i^{(1)} \right)^2 = \triangle^{(1)} = \min \tag{3.14.4}$$

Schritt $(n) \to (n+1)$:

Falls $f(x_i)^{(1)} > 0 \forall\, i$

$\Rightarrow$Verfahren beendet

sonst:

$$y_i^{(n+1)} := \begin{cases} f(x_i)^{(n)} & , \text{ falls } y_i^{(n)} \leq 0 \text{ und } f(x_i)^{(n)} < 0 \\ y_i^{(n)} & , \text{ sonst} \end{cases} \tag{3.14.5}$$

$$D_0^{(n+1)} := \left\{ \left(x_i, y_i^{(n+1)} \right) \mid y_i^{(n+1)} \leq 0 \right\} \tag{3.14.6}$$

$$D_>^{(n+1)} := \left\{ \left(x_i, y_i^{(n+1)} \right) \mid y_i^{(n+1)} > 0 \right\} \tag{3.14.7}$$

$$\begin{aligned} \Rightarrow \quad & \underbrace{\sum_{(x_i,y_i)\in D_0^{(n+1)}} \left(f(x_i)^{(n)} - y_i^{(n+1)} \right)^2}_{=0} \\ & + \sum_{(x_i,y_i)\in D_>^{(n+1)}} \left(f(x_i)^{(n)} - y_i^{(n+1)} \right)^2 \\ & = \triangle'_{(n)} \leq \triangle_{(n)} \end{aligned} \tag{3.14.8}$$

Fall $\triangle'_{(n)} = \triangle_{(n)}$:

$\triangle'_{(n+j)} = \triangle_{(n+j)} \, \forall j \geq 1 \Rightarrow$Konvergenz

Fall $\triangle'_{(n)} < \triangle_{(n)}$:

$$\begin{aligned} f(x)^{(n+1)} := \left\| f(x)^{(n+1)} - y^{(n+1)} \right\|_2^2 = \min, \\ (x,y) \in \left\{ D_0^{(n+1)} \cup D_>^{(n+1)} \right\} \end{aligned} \tag{3.14.9}$$

An den Stützstellen $\{x_i\}$ gilt:

$$\sum_{(x_i,y_i)\in D_0^{(n+1)} \cup D_>^{(n+1)}} \left(f(x_i)^{(n+1)} - y_i^{(n+1)} \right)^2 = \triangle_{(n+1)} = \min \tag{3.14.10}$$

$$\triangle_{n+1} = \min \quad \Rightarrow \quad \triangle_{(n+1)} \leq \triangle'_{(n)} \tag{3.14.11}$$

Fall $\triangle_{(n+1)} = \triangle'_{(n)}$:

Gleichung 3.14.8 $= \min \Rightarrow$Konvergenz

Fall $\triangle_{(n+1)} < \triangle'_{(n)}$:

$$\triangle_{(n+1)} < \triangle'_{(n)} < \triangle_{(n)} \tag{3.14.12}$$

Wegen $\triangle_{(j)} \geq 0$ und $\triangle'_{(j)} \geq 0 \forall j$

$$\Rightarrow \triangle_{(n)} \geq 0 \forall n \tag{3.14.13}$$

$\Rightarrow \triangle_{(n)}$ monoton fallend und nach unten beschränkt.

$\Rightarrow \lim_{n\to\infty} \triangle_{(n)} = \triangle_{min} \geq 0$

3.14.6 Beispiel Polynom 3. Ordnung

Abbildung 3.10 zeigt ein Beispiel für die sukzessive Verbesserung der Anpassung an ein Polynom 3. Ordnung mit Iterationsschritten $t = 0, ..., 5$.

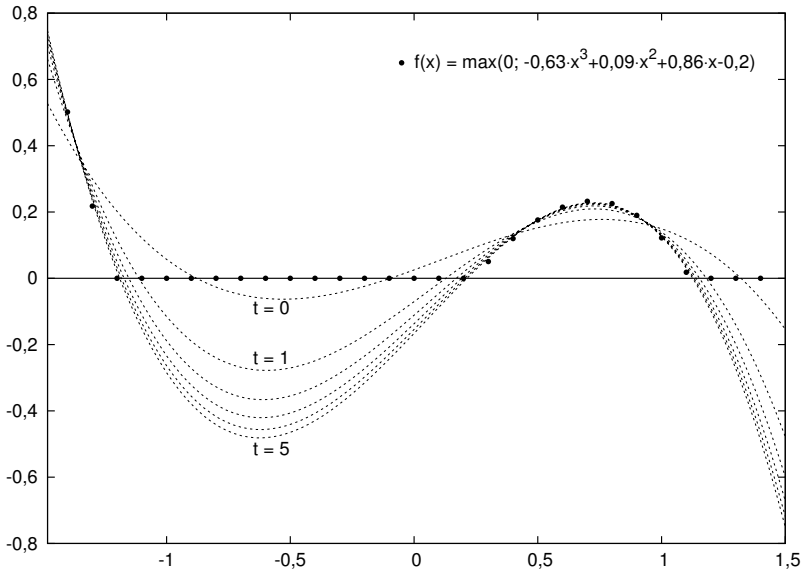


Abbildung 3.10: *Ursprünglicher Fit und die ersten fünf Iterationsschritte an ein Polynom mit Schwellwert 0.*

3.15 Ensemble aus Einzelmodellen

Mehrere auf den gleichen aber unterschiedlich auf Trainings- und Testmengen verteilten Daten adaptiv generierte Modelle bilden die gleichen globalen Zusammenhänge der Datensätze ab. Sie können allerdings zu unterschiedlichen Termstrukturen führen, da die zu minimierende Fehlerfunktion auf den jeweils unterschiedlich permutierten Ensembles aus Testfehlern unterschiedlich aussieht. Daher werden die Vorhersagen verschiedener Modelle unterschiedlich sein. Je größer die Datenmenge ist, desto gleichmäßiger

kann die Information auf die einzelnen Trainings- und Testdatensätze verteilt werden und desto geringer werden die Unterschiede ausfallen.

Die Adaptionspolynome entwickeln sich innerhalb der Menge aller zu gegebener maximaler Ordnung zur Auswahl stehenden Polynome gegen Lösungen mit lokal minimalem Fehler auf den Testdaten. Die zufälligen Schwankungen, mit denen die einzelnen Modelle um die Zielwerte streuuen, können verkleinert und damit die Vorhersagegenauigkeit verbessert werden, indem über das Ergebnis der Einzelmodelle gemittelt wird [110]. Abbildung 3.11 zeigt den Verlauf des Bestimmtheitsmaßes bei Mittelung der Fit-Daten eines Ensembles aus Einzelmodellen für ein Oszillationsmarken-Problem mit 76 Messungen und 14 Eingangsgrößen, zu dem ein adaptives Polynommodell mit Termen bis zur Ordnung 2 erstellt wurde.

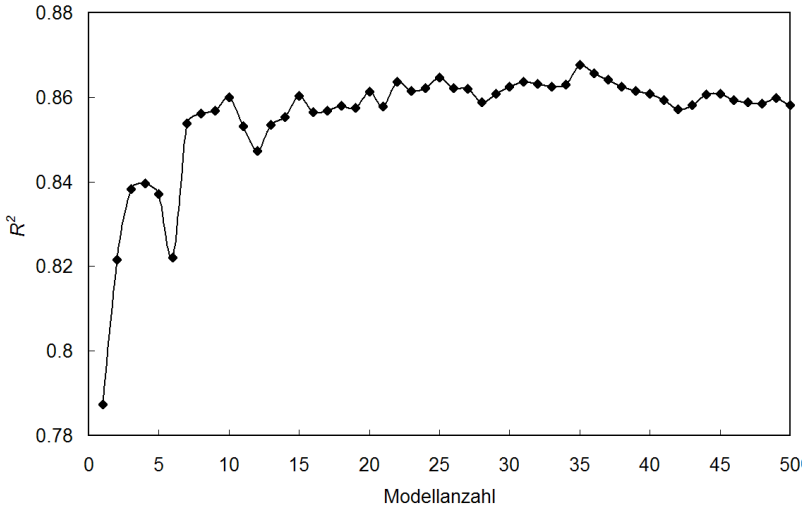


Abbildung 3.11: *Bestimmtheismaß bei Mittelung über mehrere Modelle. Ab sieben Modellen liefert die Mittelung deutlich bessere Werte als das ursprüngliche Einzelmodell.*

Wenn die Modelle zur Extrapolation benutzt werden sollen, kann die Streuung der Einzelmodelle als Maß für den Extrapolationsfehler angesehen werden. Für Anpassungen innerhalb der Datencluster kann erwartet werden, dass die Einzelstreuungen gering sind, falls die Modelle eine gute Anpassungsfähigkeit haben. Je weiter außerhalb der Messdatenbereiche Vorhersagen gemacht werden, desto mehr laufen die Modellergebnisse verschiedener Modelle aufgrund der Extrapolationsunsicherheit auseinander, da die Funktionsverläufe der Modelle in diesen Bereichen nicht mehr zu den Messwerten hin gezogen und gebündelt wurden. Zur Bewertung kann der Mittelwert der Streuungen der Vorhersagen der Einzelmodelle an allen Messdaten berechnet und mit der Streuung des neu zu berechnenden Punkts verglichen werden. Ist die Streuung deutlich stärker, dann muss die Vorhersage als unsicher betrachtet werden. Im Umkehrschluss ist eine geringe Streuung aber keine Garantie für eine realistische Vorhersage, da auch der Fall eintreten kann, dass alle Modelle aufgrund unzureichender Daten den gleichen Extrapolationsfehler

erzeugt haben.

Eine hohe Anzahl an Modellen stabilisiert weiterhin die Extrapolationsfähigkeit, da sich die zufälligen Schwankungen der Einzelmodelle teilweise entgegenwirken können.

3.16 Einbinden neuer Messergebnisse

Wenn zu einem Produktionsdatensatz neue Messdaten hinzugefügt werden, nachdem ein adaptives Polynommodell erstellt wurde, kann ein komplett neues Modell erstellt werden, um den Informationszuwachs zu berücksichtigen. Je nach Anzahl und Komplexität der Daten kann dies aber zeitaufwändig sein, so dass alternative Methoden zur Modellaktualisierung gefragt sind.

Eine Möglichkeit besteht darin, die bereits gefundene Polynomstruktur des alten Modells beizubehalten und lediglich die Regressionskoeffizienten neu anzupassen. Damit ist nur eine QR-Zerlegung nötig, was bei den hier behandelten Problemen auf einem durchschnittlichen Desktop-PC üblicherweise deutlich unter einer Sekunde dauert. Wenn nur verhältnismäßig wenig Daten neu hinzugefügt werden, ist auch nicht zu erwarten, dass zur adäquaten Abbildung eine grundsätzlich andere Polynomstruktur nötig ist.

Wenn dagegen eine größere Anzahl hinzugefügter Messungen berücksichtigt werden soll, ist die bisherige Polynomstruktur möglicherweise nicht komplex genug, um der erhöhten Informationsmenge Rechnung zu tragen und folglich wäre eine Neuanpassung der Koeffizienten ineffektiv. In dem Fall kann eine Adaption der Polynomstruktur ausgehend von der Struktur des alten Modells gemacht werden, da die Vermutung nahe liegt, dass das alte Modell zumindest teilweise im neuen Modell enthalten ist. Die Adaptionsprozedur konvergiert dann üblicherweise schneller gegen ein neues optimales Modell als eine komplette, beim konstanten Modell beginnende Anpassung.

Erst wenn die Anzahl der neu hinzugefügten Messdaten sehr hoch ist oder einer deutlich anderen Verteilung entspricht als die ursprüngliche Datenmenge, sollte eine komplette Neuerstellung der Polynomstruktur durchgeführt werden.

3.17 Auswertungsbeispiele

Die Modelle erlauben eine Abschätzung der modellierten Eigenschaft für beliebige Arbeitspunkte $\{x_1, ..., x_m\}$, wobei bei der Vorhersage noch nicht durch Messdaten gesicherter Bereiche mit Extrapolationsproblemen gerechnet werden muss (Abschnitt 3.15). Mithilfe dieser Modellvorhersagen können Optimierungen unter Nebenbedingungen gerechnet werden, um einen Arbeitspunkt zu verbessern. Beispielsweise kann eine Minimierung des HIC-Risikos als Minimierung des Modellwerts *CAR* formuliert

werden mit den Nebenbedingungen, dass die Modellwerte Streckgrenze und Zugfestigkeit sich nicht verschlechtern und vorgegebene Min- und Max-Grenzen für die Legierungselemente eingehalten werden. Dabei können noch zusätzliche Bedingungen wie die Minimierung der Produktionskostenfunktion oder das Einhalten weiterer mechanisch-technologischer Kenngrößen gestellt werden.

3.17.1 Variation um einen Arbeitspunkt

Um die Modellvorhersagen mit den erwarteten Änderungen bei Variation eines Prozessparameters x_i zu vergleichen, die beispielsweise aus Erfahrungswerten oder Untersuchungen aus der Literatur vorliegen, kann die Modellvorhersage $M(x)$ für konstante $\{x_1, ..., x_{i-1}, x_{i+1}, ..., x_m\}$ und innerhalb eines Intervalls $[x_{i,min}; x_{i,max}]$ variertem x_i aufgetragen werden. Abbildung 3.12 zeigt die Modellfunktion der Zugfestigkeit R_m für zwei Variationen des Parameters C im Intervall [300 ppm; 600 ppm] um zwei unterschiedliche Arbeitspunkte, d. h. solche mit unterschiedlichen Blechdicken, Legierungen, Gieß-, Walz-, und Kühlparametern. Während der Variation von C wurden die die restlichen Parameter jeweils konstant gehalten.

Währen sich die Zugfestigkeit bei Variation von C um Arbeitspunkt 1 näherungsweise linear erhöht, ist der Verlauf um Arbeitspunkt 2 gekrümmt. Bei geringem C-Gehalt bewirkt eine Erhöhung von C also um diesen Arbeitspunkt herum einen stärkeren Gewinn an Zugfestigkeit als bei hohem C-Gehalt. Es gibt also laut Modell keine globale, d. h. an allen Arbeitspunkten gleiche Änderung $\partial R_m/\partial \mathrm{C}$, sondern einen Aufgrund der Wechselwirkung mit anderen Prozessparametern lokal unterschiedlichen Verlauf.

Zusätzlich sind in Abbildung 3.12 weitere Messwerte von unterschiedlichen Arbeitspunkten aus der Datenbank eingetragen. Die Streuung der Punkte resultiert hauptsächlich aus der Projektion des 19-dimensionalen Parameterraums in die $R_m(\mathrm{C})$-Ebene. Ein weiterer Teil resultiert aus der Streuung der Messung und dem Anpassungsfehler (Abschnitt 1.7.1). Datencluster, die ähnliche Auftragsqualitäten darstellen, sind deutlich zu erkennen.

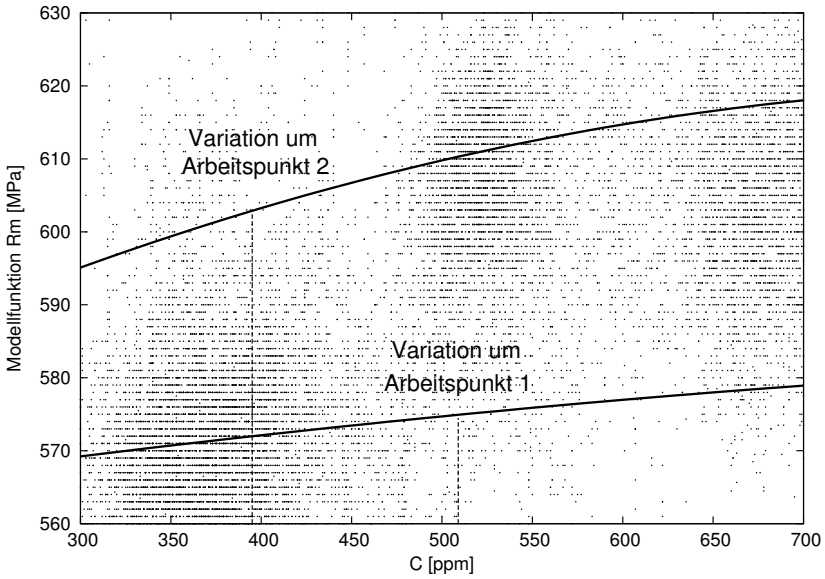


Abbildung 3.12: *Vorhersage eines adaptiven Polynommodells für Rm unter Variation von* C *um zwei verschiedene Arbeitspunkte. Weitere Arbeitspunkte aus der Datenbank sind zum Vergleich mit eingetragen.*

3.17.2 Ableitungshistogramme

Oft stellt sich beim Entwurf neuer oder Weiterentwicklung bestehender Stahlqualitäten die Frage, ob sich ein bestimmter Parameter x_i global oder zumindest in einem lokalen Bereich qualitativ gleich auf die gewünschte Materialeigenschaft auswirkt.

Als Beispiel sei die Wirkung des Legierungselements Mn auf R_m gezeigt. Hierzu kann eine bezüglich der bisherigen Produktion globale Aussage gemacht werden, indem für jeden Arbeitspunkt die Ableitung der Modellfunktion $\partial R_m / \partial \mathrm{Mn}$ berechnet und diese in einem Histogramm dargestellt wird. Daraus lassen sich statistische Größen wie etwa minimale, maximale und mittlere Wirkung abschätzen. Abbildung 3.13 zeigt die Verteilungen für Mn und Mo bei Anwendung eines R_m-Modells auf einen Produktionsdatensatz.

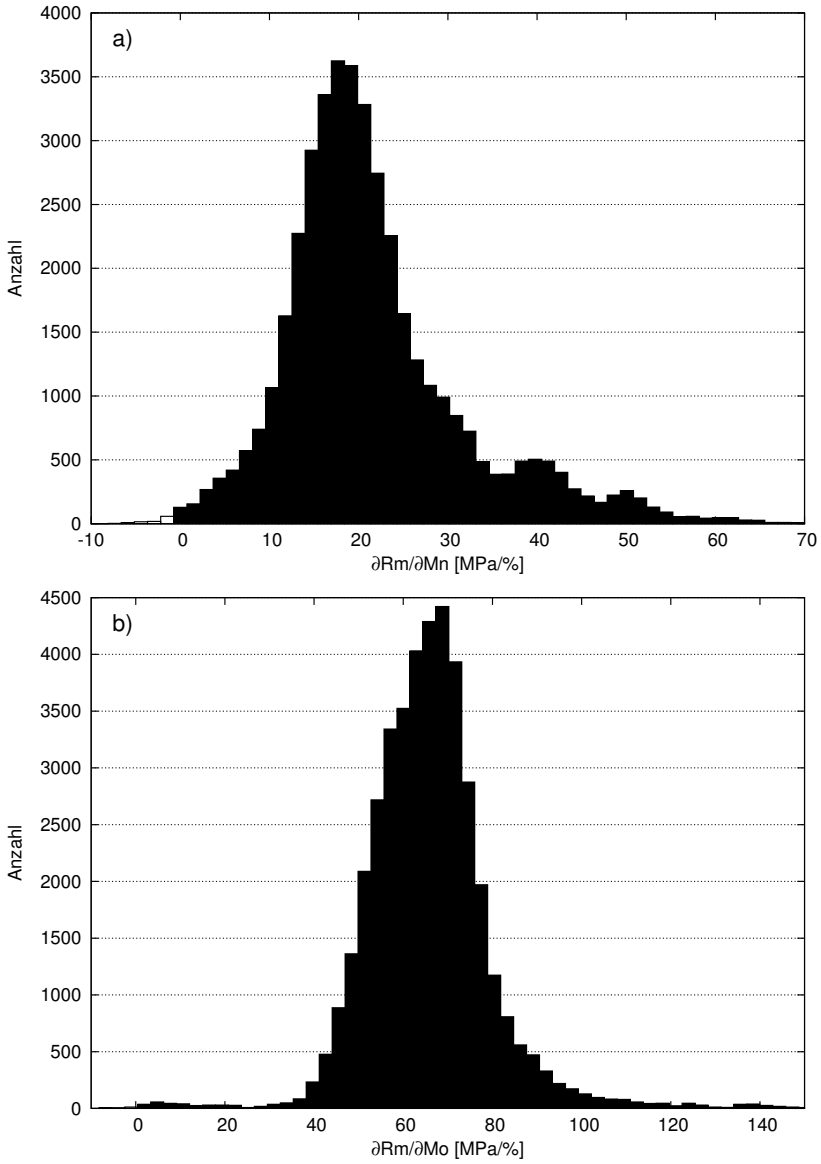


Abbildung 3.13: *Histogramme der Ableitungen der Modellfunktion a)* $\partial R_m/\partial$Mn *und b)* $\partial R_m/\partial$Mo *für alle Arbeitspunkte aus einem Produktionsdatensatz.*

Eine Erhöhung von Mn hat demnach im Modell an fast allen Arbeitspunkten eine Erhöhung von R_m zur Folge und bestätigt damit die festigkeitserhöhende Wirkung von Mn. Eine Erhöhung von Mo erhöht R_m pro legiertem Prozentpunkt stärker. Diese Aussage bezieht sich nur auf die ausgewerteten Arbeitsbereiche, bei denen Mo üblicherweise deutlich niedriger legiert ist als Mn.

Abbildung 3.14 zeigt die Auftragung für Nb. Je nach Arbeitspunkt kann dieses Element im Modell einen festigkeitserhöhenden oder -erniedrigenden Effekt haben.

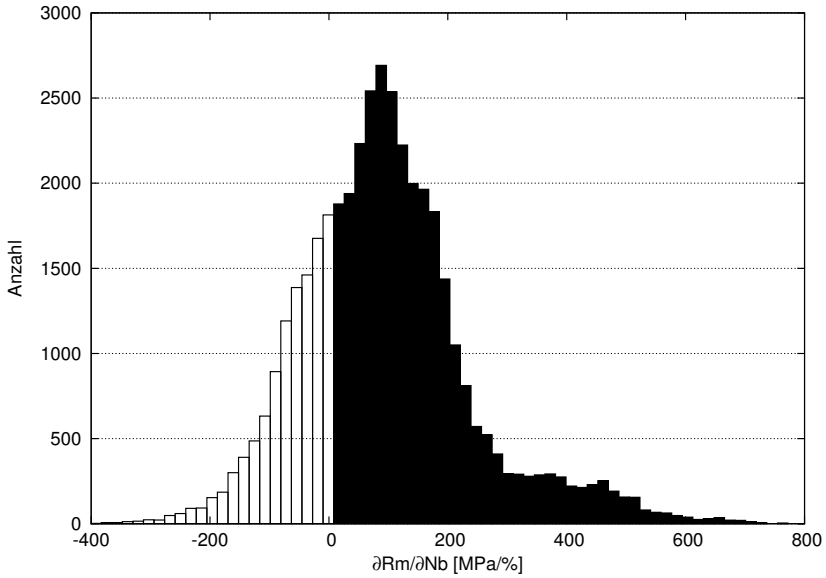


Abbildung 3.14: *Histogramm der Ableitung der Modellfunktion* $\partial R_m/\partial$Nb *für alle Arbeitspunkte aus einem Produktionsdatensatz. Die weißen Balken beinhalten die Arbeitspunkte, bei denen* Nb *eine festigkeitserniedrigende Wirkung hat, die schwarzen diejenigen mit festigkeitserhöhender Wirkung.*

Nb bewirkt eine Ausscheidungshärtung und hemmt das Kornwachstum, wodurch eine Kornfeinung und damit eine Festigkeitserhöhung erreicht wird. Wenn allerdings beim Aufheizen der Bramme keine ausreichend hohe Temerpatur und Heizdauer erreicht wird, besteht gerade bei hohen Nb-Konzentrationen die Gefahr, dass nicht das gesamte in Form von Karbiden und Nitriden vorliegende Nb in Lösung geht und somit große Nb(C,N)-Partikel in der Matrix verbleiben, die sich negativ auf die Festigkeit auswirken können. Abbildung 3.15 zeigt die Auftragung der Ableitungswerte gegen den Nb-Gehalt. Der Trend zeigt, dass die festigkeitssteigernde Wirkung von Nb mit wachsendem Nb-Gehalt im Mittel abnimmt und die negativen Ableitungswerte hauptsächlich bei hohen Nb-Gehalten liegen. Diese Vermutung zu unter bestimmten Bedingungen festigkeitssenkenden Wirkung von Nb wird daher vom Modell wiedergegeben.

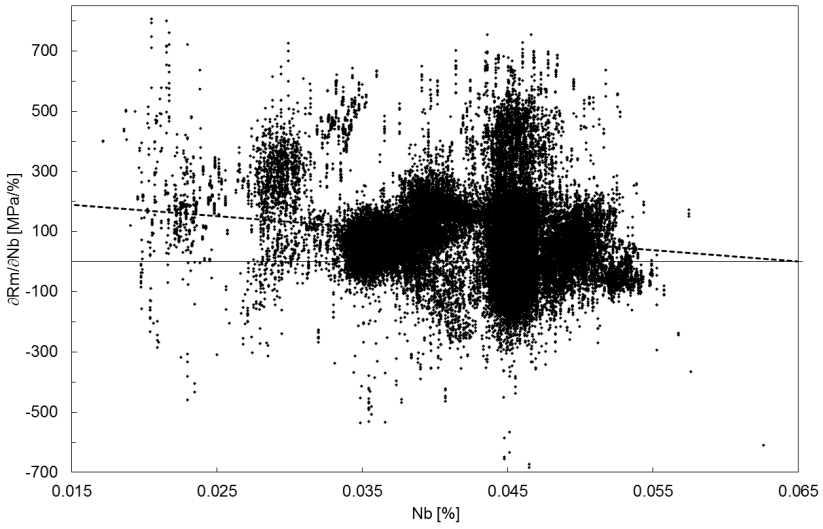


Abbildung 3.15: $\partial R_m/\partial$Nb *gegen* Nb. *Die Trendline ist gestrichelt eingezeichnet.*

3.17.3 Optimierung

Die Modellfunktionen können zur Optimierung von bereits realisierten Arbeitspunkten verwendet werden. Dazu werden für alle relevanten Eigenschaften separate Modelle erstellt und ein Optimierer angewendet, der entweder eine Eigenschaft direkt oder eine aus mehreren Eigenschaften formulierte Bedingung minimiert oder maximiert.

Ein Praxisbeispiel ist die Variation der Legierungselementgehalte, die zu einer Reduzierung der HIC-Anfälligkeit führen soll. Da dies durch eine Vielzahl an Variationen realisiert werden kann, sollen Nebenbedingungen gestellt werden, die dafür sorgen, dass andere Eigenschaften innerhalb der Toleranzbereiche gehalten werden. Das HIC-Risiko kann z. B. ausgehend von bestimmten Arbeitspunkten durch Verringerung des Nb- und Ti-Gehalts gesenkt werden. Dadurch sinkt allerdings auch die Festigkeit, was unerwünscht ist. Daher sollen alternative Arbeitspunkte gefunden werden, die den Festigkeitsverlust durch Erhöhung anderer festigkeitssteigernder Legierungselemente, die das HIC-Risiko weniger stark erhöhen, auffangen. Die Optimierungsbedingung lautet dann:

Minimiere CAR unter den Nebenbedingungen, dass

- $R_{m,min} \leq R_m \leq R_{m,max}$
- $R_{t05,min} \leq R_m \leq R_{t05,max}$
- $R_{t05,min}/R_m \leq Vorgabe$
- $Element_{i,min} \leq Element_i \leq Element_{i,max}$ (Vorgaben Analysespanne)

- $\sum Element_i \leq Vorgabe$ (Summenvorgaben, z. B. C-Äquivalent)

- $\sum (x - x_{i,0})^2 \leq r$ (Radius-Bedingung)

Das somit formulierte Optimierungsproblem ist von niedriger Dimension und stellt keine besonderen Anforderungen an den Optimierungsalgorithmus. Hierfür wurde der Optimierer des Softwarepakets AMPL [111] benutzt.

Die Radius-Bedingung beschränkt die Summe der Quadrate der Änderungen aller Parameter, so dass nur kleine Schritte in eine geeignete Verbesserungsrichtung gemacht werden. Würden zu große Schritte erlaubt und alle Parametervariationen unabhängig voneinander voll ausgeschöpft, selbst wenn die Variation bestimmter Parameter nur geringe Wirkung zeigt, bestünde die Gefahr eines zu großen Extrapolationsschritts. Statt dessen soll nur ein kleiner Schritt in eine Richtung gemacht werden, in der eine große Wirkung erzielt wird. Damit die Radiusbedingung gestellt werden kann, muss ein Abstandsmaß definiert werden, da die Parameter im Allgemeinen unterschiedliche physikalische Dimensionen haben. Hierzu kann z. B. auf die Produktionsintervalle $[x_{i,min}, x_{i,max}]$ oder Quantile davon normiert werden. Abbildung 3.16 zeigt das Ergebnis eines solchen Optimierungsdurchlaufs ausgehend von einem Arbeitspunkt mit erhöhtem HIC-Risiko, bei dem der CAR-Wert laut Modell von 1,3% auf 0% gesenkt wurde.

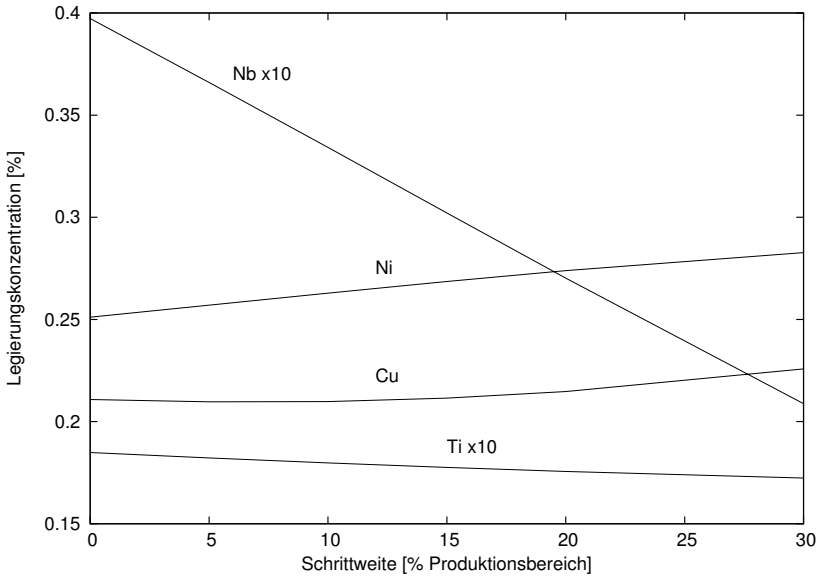


Abbildung 3.16: *Vorschlag des adaptiven Polynommodells zur Minimierung des CAR-Werts unter Nebenbedingungen durch Variation der Legierungsmittelkonzentrationen.*

3.17.4 Anwendungen in laufender Produktion

Die Vorhersagen der Modelle bei Variation eines Parameters um einen bestimmten Arbeitspunkt herum können mit Vermutungen aus metallurgischen und physikalischen Betrachtungen verglichen werden, um aus der Produktionserfahrung vermutete Zusammenhänge zu bestätigen. Die Fähigkeit der Modelle, nicht nur solche qualitativen, sondern auch quantitative Aussagen zu machen, hat sich in der Verbesserung der Produktentwicklung und Produktionsprozesse als nützliches Hilfsmittel erwiesen.

Im Produktionsalltag der Dillinger Hütte werden sowohl adaptive Polynommodelle, als auch neuronale Netze genutzt, um die Planung von Aufträgen zu unterstützen und auf neue Produktqualitäten hinzuarbeiten. Hierzu existieren Modelle zur Vorhersage mechanisch-technologischer Eigenschaften für verschiedene Produktionsrouten.

Ebenso kommt die Frage auf, welche Korrekturen während der laufenden Produktion gemacht werden können, um die Treffsicherheit der vorgegebenen Materialeigenschaften zu erhöhen und somit das Ausfallrisiko zu senken. Die Produktion ist dabei schon so weit fortgeschritten, dass nur noch ein Teil der Parameter geändert werden können, beispielsweise wenn die Brammen bereits vergossen wurden, so dass weder die chemische Zusammensetzung, noch das Ausgangsformat geändert werden können. Dann kann die Frage etwa lauten, wie stark die Endkühltemperatur abgesenkt werden muss, um die Festigeit des Materials um einen bestimmten Betrag zu erhöhen und ob andere Materialeigenschaften, z. B. die Zähigkeit, sich dadurch unzulässig stark verschlechtern. Dies kann durch Anwendung von Polynommodellen abgeschätzt und mit Erwartungen aus metallurgischer Erfahrung verglichen werden. Auf diese Weise konnten die Modelle bereits erfolgreich zur Verbesserung von Aufträgen mit sehr anspruchsvollen mechanisch-technologischen Vorgaben beitragen.

Kapitel 4

Adaptive künstliche neuronale Netze

4.1 Strategie

Die Grundlagen künstlicher neuronaler Netze sind im Appendix (Abschnitt 7.2) dargestellt. Die Bestimmung einer dem Problem angepassten Netztopologie ist ein wichtiger Bestandteil der Modellbildung. Rein stochastisch operierende genetische Verfahren müssen zur Topologieoptimierung viele Kandidatennetze auswerten. Da die Optimierung eines einzelnen Kandidaten bereits relativ rechenaufwändig sein kann, sind solche Ansätze nur auf Probleme mit wenigen Eingangsgrößen und Messwerten anwendbar. Daher wird im Folgenden ein neues eigenes Verfahren zur schrittweisen Entwicklung von einfachen zu komplexeren Netzen vorgestellt. Die Vorgehensweise ist der schrittweisen Polynomentwicklung (Abschnitt 3.2) ähnlich.

4.2 Erweiterung der Topologie

Es wird die maximale Netztopologie vorgegeben, im Folgenden ein Feedforward Multi Layer Perceptron (MLP, Abschnitt 7.2.2) mit L Ebenen und $N(l=1, ...,L)$ Neuronen pro Ebene. Diese Topologie wird mit $N(1)$-$N(2)$- ...-$N(L)$ bezeichnet. Der Algorithmus soll ausgehend von einer minimalen Anzahl Verbindungen aus der Menge aller noch nicht verwendeten Verbindungen schrittweise eine geeignete Topologie aufbauen. Dazu sollen dem momentanen Kandidatennetz in jedem Iterationsschritt nacheinander alle noch nicht verwendeten Verbindungen probeweise einzeln hinzugefügt, trainiert und bewertet werden. Nicht verwendete Verbindungen werden innerhalb des vollständigen Netzes realisiert, indem deren Gewichte Null gesetzt werden und während des Lernvorgangs nicht verändert werden dürfen. Wenn Verbindungen hinzugefügt werden, die keine Auswirkung auf das Ausgabeneuron haben können, dann betrachten wir das Netz als unvollständig. Das kann in folgenden Fällen passieren:

1. Es wird zu einem Neuron hin verbunden, dessen Ausgabe nicht direkt oder über Neuronen in den Folgeebenen mit dem Ausgabeneuron verbunden ist. In diesem Fall wird anstatt des unvollständigen Kandidatennetzes die Menge aller möglichen Kandidatennetze erstellt, in der die Ausgabe des Neurons über jeweils eine zusätzliche Verbindung pro folgender Schicht erweitert wird, bis das Netz vollständig ist.

2. Es wird von einem Neuron aus verbunden, dessen Eingabe noch nicht verbunden ist. In diesem Fall wird anstatt des unvollständigen Kandidatennetzes die Menge aller möglichen Kandidatennetze erstellt, in denen das Neuron mit jeweils einer zusätzlichen Verbindung pro Vorgängerschicht verbunden wird, bis das Netz vollständig ist.

Der Algorithmus aktiviert zunächst eine noch nicht verwendete Verbindung. Falls diese neu entstandene Topologie vollständig ist, wird dieses Netz als einziger Kandidat verwendet. Falls die Topologie unvollständig bezüglich der Eingangsseite ist, werden weitere Teilnetze erstellt, indem vom Anfangsneuron der neuen Verbindung aus jede eingehende Verbindung separat aktiviert wird. Alle diese neu generierten Netze werden ebenfalls auf Vollständigkeit in Richtung der eingehenden Verbindungen überprüft (Abbildung 4.1). Das Hinzufügen neuer Verbindungen wird für alle unvollständigen Teilnetze bis zur Vollständigkeit in Richtung der Eingangsseite rekursiv wiederholt. Die gleiche Erweiterungsprozedur wird ausgehend vom Endneuron analog für die Ausgangsseite durchgeführt. Die Menge der vollständigen Kandidatennetze ergibt sich dann durch Kombination aller vervollständigten Netze der Ein- und Ausgangsseite. Jedes Kandidatennetz enthält dann maximal eine neue Verbindung zwischen je zwei Schichten. Die Verbindungen zu Bias-Neuronen werden für alle verbundenen Neuronen aktiviert.

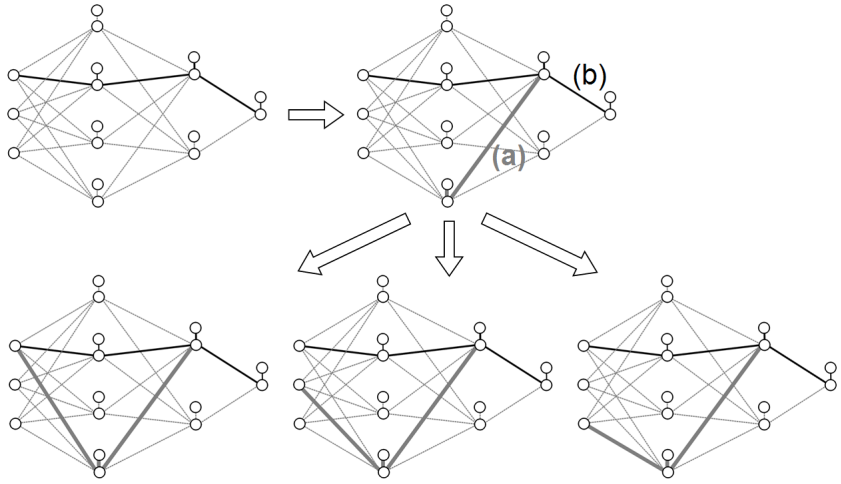


Abbildung 4.1: *Erweiterung der Netztopologie: zu einem bestehenden Netz (durchgezogene Verbindungen) wird die Verbindung (a) und das Bias-Neuron auf der Eingangsseite hinzugefügt. (a) ist über die bereits bestehende Verbindung (b) vollständig in Richtung der Ausgabeebene, jedoch unvollständig in Richtung der Eingabeebene. Der Algorithmus vervollständigt die Topologie durch die Menge aller Netze mit einer weiteren Verbindung pro Vorgängerschicht, durch die (a) vollständig wird.*

Beim probeweisen Hinzufügen aller noch nicht verwendeten Gewichte können durch die Fälle (1.) und (2.) identische Netze mehrfach generiert werden. Das sollte im Algorithmus überprüft werden, damit diese nicht mehrfach ausgewertet werden. Durch eine Speicherung des Fehlers zu einer eindeutigen Kennzeichnung der Netztopologie wird vermieden, dass auch in zukünftigen Iterationsschritten identische Auswertungen wiederholt werden.

4.3 Vereinfachung der Topologie

Nach einer vorgegebenen Anzahl Iterationen sollte das Netz vereinfacht werden, um im Verlauf der Topologieentwicklung überflüssig gewordene „Kompromissgewichte" (analog der Kompromissterme beim Polynommodell, Abschnitt 3.5) zu entfernen und das Netz robuster gegen Überanpassung zu machen. Sollte der Testfehler des vereinfachten Netzes geringer sein als der des ursprünglichen Netzes, wird ein weiterer Vereinfachungsschritt durchgeführt. Beim Entfernen einer Verbindung wird überprüft, ob Folgeneuronen dadurch unvollständig verbunden werden. In diesem Fall werden alle ausgehenden Verbindungen dieses Neurons ebenfalls gelöscht und auf weitere unvollständige Verbindungen überprüft.

Algorithmus 4.1 zeigt den Pseudocode zur adaptiven Entwicklung eines MLPs.

Algorithmus 4.1 *Pseudocode der adaptiven MLP-Entwicklung.*

Erstelle D_{test} und D_{train} aus D (Gl. 1.6.5)
$Modell^{(1)}$ = *MLP*, alle Verbindungen deaktiviert

$n = 1$
iteriere, bis Abbruchkriterium erfüllt
  ERWEITERN: für $j = 1, ..., \delta^+$:
    $n = n + 1$
    Für alle inaktiven Verbindungen $V \in$ *MLP*:
      Modellkandidat: Aktiviere V in $Modell^{(n-1)}$
      alle Verbindungen in *Modellkandidat* vollständig?
        nein: vervollständigen
      Training *Modellkandidat* auf *Trainingsdaten*
      Fehlerbewertung *Modellkandidat* bezüglich *Testdaten*
    $Modell^{(n)}$ = *Modellkandidat* mit geringstem Fehler
  REDUZIEREN: für $j = 1, ..., \delta^-$:
    $n = n + 1$
    Für alle aktiven Verbindungen $V \in Modell^{(n-1)}$:
      Modellkandidat: Deaktiviere V in $Modell^{(n-1)}$
      Modellkandidat enthält unvollständige Verbindungen?
        ja: unvollständige Verbindungen löschen
      Training *Modellkandidat* auf *Trainingsdaten*
      Fehlerbewertung *Modellkandidat* bezüglich *Testdaten*
    $Modell^{(n)}$ = *Modellkandidat* mit geringstem Fehler
  Wenn Fehler($Modell^{(n)}$) $<$ Fehler($Modell^{(n-1)}$) :
    REDUZIEREN

Für jede Topologieoptimierung wird eine Mindestanzahl δ^+ an Iterationsschritten vorgegeben, in denen das Netz erweitert wird und eine Mindestanzahl $\delta^- < \delta^+$, in denen es vereinfacht wird. Da die neu entstehenden Netze bei Hinzufügen und Entfernen von Gewichten immer vervollständigt werden, kann sich die Anzahl der Gewichte in einem einzelnen Schritt um mehr als eins ändern. Falls sich die Anpassungsgüte nach Entfernen einer Verbindung weiter verbessert, wird so lange ein weiterer Entfernungsschritt angehängt, bis keine weitere Verbesserung mehr stattfindet. Wenn durch Hinzufügen und Entfernen von Verbindungen die gleiche Topologiefolge wiederholt wird, kann das Training abgebrochen oder δ^+ erhöht werden, um zu versuchen, zu einem anderen Topologieoptimum zu gelangen.

4.4 Lern- und Bewertungsschema

Die erste auszuwertende Population von Netzkandidaten ist die Menge aller Netze, die genau eine Verbindung zwischen den Ebenen enthält. Die Netze werden trainiert und das Netz mit dem minimalen Fehler auf den Testdaten wird beibehalten. Diese einfachen Netze zu Beginn der Topologieiteration können schnell und aufgrund der einfachen Fehlerfunktion robust trainiert werden. Das Training mittels Backpropagation kann abgebrochen werden, sobald der Testfehler sein erstes Minimum durchlaufen hat. Es ist zwar möglich, dass nach weiteren Iterationsschritten ein besseres Minimum gefunden wird (Abbildung 4.2), aber diese vollständige Suche ist wegen des hohen Zeitaufwands nicht praktikabel, da ohnehin nicht garantiert werden kann, dass ein globales Minimum gefunden wird. Wenn Rechenzeit kritisch ist und eine ungenauere Bewertung der Kandidatennetze in Kauf genommen wird, kann auch für jedes Netz eine maximale Anzahl an Iterationsschritten vorgegeben werden unter der Annahme, dass das Netz, das innerhalb dieser Zeit den geringsten Fehler erreicht, die am besten geeignete Topologie darstellt. Das beste Netz wird das Ausgangsnetz für den nächsten Iterationsschritt.

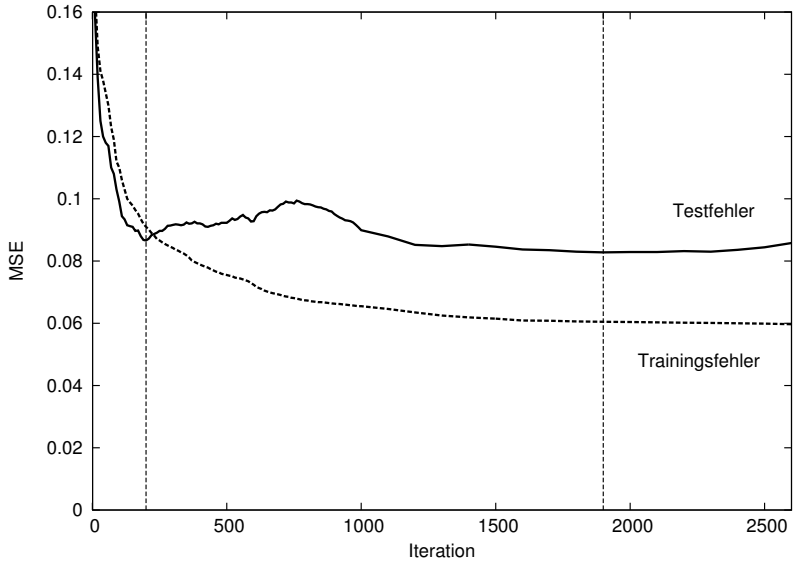


Abbildung 4.2: *Backpropagation-Fehlerverlauf am Beispiel von HIC-Daten, bei dem nach 200 Iterationen das erste Minimum des Testfehlers durchlaufen wird, gefolgt von einem besseren nach 1900 Iterationen.*

Um die Trainingszeit zu verkürzen, kann die Information eines Optimums an das Netz im Folgeschritt vererbt werden, indem die trainierten Gewichte des Ursprungsnetzes zur Initialisierung des Folgenetzes verwendet werden. Diese Vererbung hat aber den Nachteil, dass ggf. neu entstehende bessere lokale Optima übersehen werden können, da die Optimierung in der Nähe des letzten lokalen Minimums startet.

Die Neuinitialisierung der Gewichte nahe Null liefert möglicherweise bessere Ergebnisse.

Um das Backpropagation-Lernen zu beschleunigen, kann anstatt der expliziten Berechnung der Aktivierungsfunktion und ihrer Ableitung in jedem Trainingsschritt eine Wertetabelle benutzt werden. Diese wird vor Iterationsbeginn erstellt und enthält die Funktionswerte an Stützstellen in Intervallen, in denen die Funktion annähernd konstant ist. Zur näherungsweisen Berechnung der Funktion in der Lernphase kann zwischen den Stützstellen interpoliert werden. Das Rechnen mit intervallweise konstanten Werten hat sich bei hinreichend feiner Intervalleinteilung allerdings schon als präzise genug erwiesen. Dadurch wurde die Iterationsdauer eines fest vorgegebenen MLPs je nach Größe des Netzes und der Wertetabelle um das fünf- bis achtfache schneller als die C++-Implementierung des $tanh(x)$ und dessen Ableitung.

4.5 Permutieren der Datensätze

Wenn aus einem Vereinfachungsschritt wieder eine bereits in einem vorherigen Iterationsschritt gewählte Topologie als bestes Netz hervorgeht, kann das Verfahren beendet werden. Alernativ kann versucht werden, eine bessere Topologie zu erreichen, indem in den Folgeschritten mit einer durch Neuverteilung der Daten auf die Test- und Trainingsmenge geringfügig geänderte Fehlerlandschaft gearbeitet wird, analog zu Abschnitt 3.8.

4.6 Capped data

Eine Beschränkung der Messwerte der Zielgröße durch Schwellwerte wie in Abschnitt 3.14 definiert, kann auch im Training neuronaler Netze implementiert werden. Als Beispiel wird wieder der Fall von Zielgrößen betrachtet, deren Verteilung bei Null abgeschnitten ist. Dazu wird in diesem Fall die Differenz aus Zielgröße $y(x_1,\ldots,x_m)_i$ und Netzausgabe $f(x_1,\ldots,x_m)_i$ bei der Berechnung der Gewichtsänderung im Backpropagation-Schritt durch die Zielgröße selbst ersetzt, falls die Netzausgabe negativ ist. Bei der Bewertung des Gesamtfehlers wird die Netzausgabe ersetzt durch

$$\bar{f}(x_1,\ldots,x_m)_i = max\{0; f(x_1,\ldots,x_m)_i\} \tag{4.6.1}$$

Dadurch wird die Anpassung an die Zielgrößenwerte $y(x_1,\ldots,x_m)_i \geq 0$ nicht durch den Versuch behindert, die Werte $y(x_1,\ldots,x_m)_j = 0$ möglichst exakt durch Null zu approximieren. Beliebige obere oder untere Schwellwerte können analog implementiert werden.

Kapitel 5

Anwendungen

5.1 Adaptives Polynommodell

5.1.1 Benchmark-Funktionen

In den folgenden Tabellen sind die Bestimmtheitsmaße der Anpassung eines adaptiven Polynommodells maximaler Ordnung q_{max} dargestellt. Der Messdatensatz D mit Eingangsparametern aus dem Intervall $[-5;5]$ wurde zur Anpassung der Kandidatenmodelle jeweils in eine Teilmenge D_{train} zur Koeffizientenanpassung aufgeteilt und anschließend über die Anpassungsgüte an die restliche Teilmenge $D_{test} = D \setminus D_{train}$ bewertet. Dies erfolgte über insgesamt $|D_{train}| \,/\, |D_{test}|$ Kreuzvalidierungsschritte, so dass die Vereinigung der in allen Kreuzvalidierungsschritten verwendeten Testmengen den gesamten Trainingsdatensatz abdeckt. Nach Abschluss des Adaptionsverfahrens wurden die Bestimmtheitsmaße der Modellvorhersagen für separate gleichverteilte Validierungsdatensätze gebildet, die dem Modell zuvor im Training nicht präsentiert wurden. R^2_{train} bezieht sich auf den Vergleich der Modellvorhersagen mit dem Trainingsdatensatz. $R^2_{valid,int}$ beschreibt die Interpolationsfähigkeit bezüglich eines zu D disjunkten Validierungsdatensatzes aus dem Trainingsintervall $[-5;5]$. $R^2_{valid,ext}$ beschreibt die Extrapolationsfähigkeit bezüglich eines Validierungsdatensatzes aus dem Intervall $[5;15]$.

Das Termerweiterungsschema während der Optimierung war in allen Fällen $\delta^+ = 2, \delta^- = 1$ (Abschnitt 3.5). Die Iteration wurde abgebrochen, sobald sich der *MSE* während des Iterationsverlaufs nicht mehr wesentlich geändert hat. Als Bewertungsmaß der Verallgemeinerungsfähigkeit der Termstrukturen wurde in jedem Schritt der über alle Kreuzvalidierungsschritte gemittelte MSE_{test} herangezogen. Als endgültige Termstruktur wurde dann diejenige mit dem geringsten MSE_{test} gewählt. Diese wurde abschließend an den gesamten Trainingsdatensatz angepasst.

Zum Vergleich wurden die Ergebnisse der Forward Selection gegenübergestellt, die schrittweise je

einen Term hinzufügten, bis sich AIC bzw. BIC nicht weiter verbessern (FS-AIC bzw. FS-BIC).

Benchmark-Funktion 1 (2D)

In Tabelle 5.1 ist die Anpassungsgüte an die zweidimensionale Benchmark-Funktion 1 (Gleichung 2.2.1 auf Seite 29) für die maximalen Ordnungen $q_{max} = 3$ und $q_{max} = 5$ dargestellt. Die Anzahl der Terme des entsprechenden vollständigen Polynoms wird mit p_{max} bezeichnet, die Anzahl der vom Modell daraus ausgewählten Terme mit p_{Modell}.

Die Anpassung höherer maximaler Ordnung liefert ein besseres Bestimmtheitsmaß auf den Trainings- und Interpolations-Validierungsdaten als die Anpassung niedriger Ordnung, schneidet aber bei den Extrapolations-Validierungsdaten schlechter ab. Die Aufteilung von D während des Adaptionsverfahrens zu $|D_{test}|$=25% oder 50% machte keinen signifikanten Unterschied.

Die hohe Anzahl unverrauschter gleichverteilter Trainingsdaten ($n = 500$) und niedrige Dimension des Problems erlauben eine solide Anpassung vollständiger Polynome. Das vollständige Polynom 3. Grades ($p_{max} = 9$) erzielte eine ähnlich gute Anpassung mit $R^2_{train} = 0,89$, $R^2_{valid,int} = 0,88$ und $R^2_{valid,ext} = 0,98$. Das vollständige Polynom 5. Grades ($p_{max} = 20$) ergab $R^2_{train} = 0,99$, $R^2_{valid,int} = 0,98$ und $R^2_{valid,ext} = 0,96$.

Tabelle 5.2 zeigt die Ergebnisse der mit Forward Selection AIC und BIC ermittelten Polynome. AIC wählt mehr Terme aus als das adaptive Polynom, BIC weniger. Die Anpassung ist mit allen Modellen vergleichbar gut.

q_{max}	$\|D_{test}\|$	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	p_{max}
3	25%	0,89	0,88	0,99	5	9
3	50%	0,89	0,88	0,99	6	9
5	25%	0,98	0,98	0,75	8	20
5	50%	0,98	0,98	0,8	9	20

Tabelle 5.1: *Anpassungsgüte des adaptiven Polynommodells an die Benchmark-Funktion 1.*

	FS-AIC				FS-BIC			
q_{max}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}
3	0,89	0,88	0,99	9	0,89	0,88	0,98	3
5	0,98	0,98	0,96	20	0,98	0,98	0,81	5

Tabelle 5.2: *Anpassungsgüte der Forward Stepwise Regression an die Benchmark-Funktion 1.*

Benchmark-Funktion 2 (10D)

Tabelle 5.3 zeigt die Anpassungsgüte an die 10-dimensionale Benchmark-Funktion 2 (Gleichung 2.2.2 auf Seite 29). Das Modell mit maximaler Ordnung 5 erzielte eine sehr gute Anpassung an den Trainings- und den Interpolations-Validierungsdatensatz und eine bessere Vorhersage für den Extrapolations-Validierungsdatensatz als das Modell maximaler Ordnung 3. Die 4-fach Kreuzvalidierung lieferte bessere Ergebnisse bei der Extrapolation als die 2-Fach Kreuzvalidierung. Die Ergebnisse auf den Extrapolations-Datensätzen schwanken im Vergleich zur Interpolation relativ stark, so dass Vergleiche zweier Ergebnisse nur dann sinnvoll sind, wenn sie sich deutlich unterscheiden.

Ein vollständiges Polynom 3. Grades ($p_{max} = 285$ Terme) lieferte zum Vergleich mit $R^2_{train} = 0,86$, $R^2_{valid,int} = 0,23$ und $R^2_{valid,ext} = 0,42$ eine schlechtere Anpassung an alle Datensätze als die adaptiven Polynome mit maximaler Ordnung $q_{max} = 5$ und $p_{modell} = 210$ bzw. $p_{modell} = 125$. Eine Anpassung eines vollständigen Polynoms vom Grad > 3 war nicht möglich, da die Anzahl der Koeffizienten hierbei die Anzahl der Trainingsdaten übersteigt.

FS-AIC (Tabelle 5.4) generierte komplexere Modelle mit geringen Abstrichen an Generalisierungsfähigkeit. Das FS-BIC-Modell mit $q_{max} = 5$ war deutlich komplexer als das adaptive Polynommodell, zeigte aber trotzdem keine Tendenz zur Überanpassung.

q_{max}	$\|D_{test}\|$	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	p_{max}
3	25%	0,7	0,32	0,35	45	285
3	50%	0,67	0,35	0,5	33	285
5	25%	1	0,99	0,93	210	3002
5	50%	1	0,99	0,65	125	3002

Tabelle 5.3: *Anpassungsgüte des adaptiven Polynommodells an die Benchmark-Funktion 2.*

	FS-AIC				FS-BIC			
q_{max}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}
3	0,86	0,23	0,48	250	0,61	0,42	0,38	17
5	0,99	0,65	0,92	250	1	0,98	0,91	500

Tabelle 5.4: *Anpassungsgüte der Forward Stepwise Regression an die Benchmark-Funktion 2.*

Benchmark-Funktion 3 (10D, verrauscht)

Tabelle 5.5 fasst die Anpassungen an die verrauschte 10-dimensionale Benchmark-Funktion 3 (Gleichung 2.2.3 auf Seite 29) zusammen. Die Vorhersage für den Test- und den Interpolations-Validierungsdatensatz

ist ähnlich wie bei der Anpassung an die unverrauschte Funktion. Zum Vergleich lieferte die Anpassung eines vollständigen Polynoms 3. Grades $R^2_{train} = 0,86$, $R^2_{valid,int} = 0,22$ und $R^2_{valid,ext} = 0,44$.

Sowohl die mit FS-AIC, als auch die mit FS-BIC generierten Modelle erzielten ebenfalls gute Anpassungsergebnisse (Tabelle 5.6). FS-AIC generierte ein komplexeres Modell mit etwas schlechterer Generalisierungsfähigkeit als FS-BIC. Das FS-BIC-Modell war weniger komplex als das adaptive Polynom und erzielte die besten Ergebnisse auf den Test- und Validierungsdaten. Es ist erwähnenswert, dass FS-BIC mit $q_{max} = 5$ bei diesem Problem ein Modell mit doppelt so vielen Parametern generierte als FS-AIC.

q_{max}	$\|D_{test}\|$	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	p_{max}
3	25%	0,71	0,33	0,69	47	285
3	50%	0,65	0,34	0,34	31	285
5	25%	1	0,97	0,58	209	3002
5	50%	1	0,98	0,81	123	3002

Tabelle 5.5: *Anpassungsgüte des adaptiven Polynommodells an die verrauschte Benchmark-Funktion 3.*

	FS-AIC				FS-BIC			
q_{max}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}
3	0,86	0,22	0,41	250	0,61	0,43	0,49	16
5	1,00	0,97	0,84	250	1	0,98	0,90	184

Tabelle 5.6: *Anpassungsgüte der Forward Stepwise Regression an die Benchmark-Funktion 3.*

Benchmark-Funktion 4 (10D, Capped)

Die 10-dimensionale Benchmark-Funktion 4 (Gleichung 2.2.4 auf Seite 29) besteht aus positiven Capped Data-Funktionswerten. Tabelle 5.7 zeigt die Bestimmtheitsmaße der Anpassungen ohne Capped Data-Schritte. Zur Berechnung des Bestimmtheitsmaßes wurden alle Modellwerte ≤ 0 durch Null ersetzt.

Die Anpassungsgüte bei höherer maximaler Ordnung ist erwartungsgemäß besser als bei niedriger Ordnung.

q_{max}	$\|D_{test}\|$	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	p_{max}
3	25%	0,8	0,44	0,03	51	285
3	50%	0,77	0,47	0,29	32	285
5	25%	1	0,77	0,41	203	3002
5	50%	0,99	0,9	0,77	99	3002

Tabelle 5.7: *Anpassungsgüte des adaptiven Polynommodells an die Benchmark-Funktion 4 ohne Capped Data-Iterationen.*

In Tabelle 5.8 sind die Ergebnisse einer Anpassung dargestellt, bei der in jedem Schritt der Termstrukturentwicklung drei Capped Data-Iterationen durchgeführt wurden wie in Abschnitt 3.14 beschrieben. Nach Abschluss der Termanpassung wurden zur Berechnung der Koeffizienten zehn Capped Data-Iterationen durchgeführt. Dieses Verfahren führte zu einer geringfügigen Verbesserung der Anpassungsgüte.

Die FS-AIC und FS-BIC Vergleichsalgorithmen wurden ebenfalls um Capped Data-Iterationen ergänzt. Abgesehen von FS-BIC mit $q_{max} = 5$ blieb die Verallgemeinerungsfähigkeit allerdings hinter der des adaptiven Polynoms zurück (Tabelle 5.9).

q_{max}	$\|D_{test}\|$	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	p_{max}
3	25%	0,88	0,47	0,12	53	285
3	50%	0,85	0,53	0,41	40	285
5	25%	1	0,86	0,62	154	3002
5	50%	1	0,91	0,68	98	3002

Tabelle 5.8: *Anpassungsgüte des adaptiven Polynommodells an die Benchmark-Funktion 4 mit Capped Data-Iterationen.*

	FS-AIC				FS-BIC			
q_{max}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}
3	0,90	0,28	0,1	250	0,64	0,58	0,24	21
5	0,95	0,27	0,17	250	1	0,93	0,78	92

Tabelle 5.9: *Anpassungsgüte der Forward Stepwise Regression an die Benchmark-Funktion 4 mit Capped Data-Iterationen.*

Benchmark-Funktion 5 (10D, Capped, verrauscht)

Bei der Benchmark-Funktion 5 wurde den Daten ein Rauschen überlagert (Gleichung 2.2.5 auf Seite 29). Tabelle 5.10 zeigt die Ergebnisse unter Berücksichtigung von Capped Data-Iterationen wie im Fall der unverrauschten Funktion. Die Anpassungsgüte verschlechtert sich durch das Rauschen insbesondere bei der Extrapolation. Zusätzlich wurde die Aufteilung in Trainings- und Testdaten bei verschiedenen Anpassungen jeweils für eine 10-fach, 4-fach und 2-fach Kreuzvalidierung varriert. Dies hatte keine signifikante Auswirkung auf die Interpolationsgüte. Die Extrapolation zeigte ein umso besseres Bestimmtheitsmaß, je weniger Terme p_{modell} im Modell verwendet wurden.

Bei verrauschten Cap-Daten ging die Verallgemeinerungsfähigkeit von FS-AIC deutlich zurück. Lediglich FS-BIC erreichte bei $q_{max} = 5$ fast die Anpassungsgüte des adaptiven Polynomalgorithmus (Tabelle 5.11).

q_{max}	$\|D_{test}\|$	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	p_{max}
3	10%	0,85	0,38	0,18	37	285
3	25%	0,88	0,37	0,06	55	285
3	50%	0,85	0,41	0,09	41	285
5	10%	1	0,81	0,3	198	3002
5	25%	1	0,8	0,3	171	3002
5	50%	1	0,84	0,7	98	3002

Tabelle 5.10: *Anpassungsgüte des adaptiven Polynommodells an die Benchmark-Funktion 5 mit Capped Data-Iterationen.*

	FS-AIC				FS-BIC			
q_{max}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}
3	0,91	0,17	0,02	250	0,80	0,44	0,09	28
5	0,95	0,17	0,04	250	1	0,82	0,59	149

Tabelle 5.11: *Anpassungsgüte der Forward Stepwise Regression an die Benchmark-Funktion 5 mit Capped Data-Iterationen.*

Benchmark-Funktion 6 (10D, Capped, verrauscht, in Clustern angeordnet)

Tabelle 5.12 zeigt die Anpassungsergebnisse an die verrauschte Benchmark-Funktion 6 (Gleichung 2.2.6 auf Seite 29), bei der zum Lernen in Clustern angeordnete Daten verwendet wurden. Die Anpassung an die Trainingsdaten ist ähnlich gut wie im Fall des Lernens mit gleichverteilten Daten, bei der Inter- und Extrapolation auf die gleichverteilten Validierungsdaten macht sich der Effekt der Clusterung jedoch in einer Verschlechterung der Anpassungsgüte bemerkbar. Lediglich bei der Entwicklung mit $q_{max} = 5$ und 2-fach Kreuzvalidierung wurde $R^2_{valid,ext} = 0,54$ erreicht.

Alle Algorithmen gaben die Lerndaten gut wieder, aber nur das adaptive Polynom mit $q_{max} = 5$ lieferte auf den Cluster-Daten akzeptable Vorhersageergebnisse. Wie auch bei den vorherigen Beispielen lieferte eine niedrigere Anzahl an Kreuzvalidierungsschritten Modelle mit besserer Verallgemeinerungsfähigkeit (Tabelle 5.13).

Abbildung 5.1 zeigt den Korrelationsplot des adaptiven Polynoms mit $q_{max} = 5$ und $|D_{test}| = 50\%$ (zwei Kreuzvalidierungsschritte).

q_{max}	$\|D_{test}\|$	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	p_{max}
3	10%	0,86	0,18	0	71	285
3	25%	0,83	0,17	0,08	47	285
3	50%	0,99	0,69	0,28	87	285
5	10%	1	0,32	0,2	156	3002
5	25%	1	0,52	0	143	3002
5	50%	0,99	0,77	0,54	87	3002

Tabelle 5.12: *Anpassungsgüte des adaptiven Polynommodells an die Benchmark-Funktion 6 (Cluster-Daten) mit Capped Data-Iterationen.*

	FS-AIC				FS-BIC			
q_{max}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}
3	0,91	0,11	0	250	0,79	0,20	0,02	36
5	0,95	0,13	0	250	0,99	0,73	0,02	93

Tabelle 5.13: *Anpassungsgüte der Forward Stepwise Regression an die Benchmark-Funktion 6 mit Capped Data-Iterationen.*

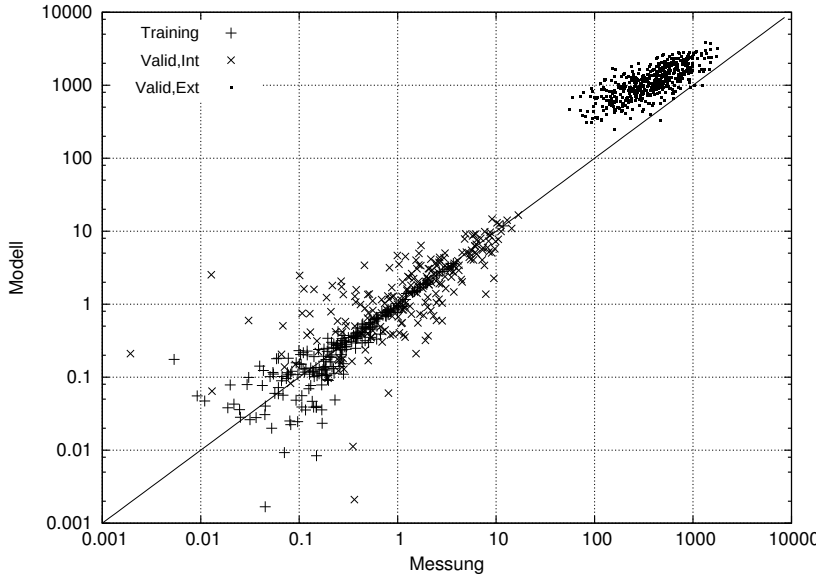


Abbildung 5.1: *Korrelationen der Vorhersage des adaptiven Polynommodells maximaler Ordnung 5 und 2-fach Kreuzvalidierung für die Datensätze Training, Interpolations-Validierung und Extrapolations-Validierung.*

Benchmark-Funktion 7 (19D, Capped, verrauscht, in Clustern angeordnet)

Tabelle 5.14 zeigt die Anpassungsergebnisse an die 19-dimensionale Benchmark-Funktion 7 (Gleichung 2.2.7 auf Seite 30), mit den Eigenschaften Rauschen, Clusterung und Capped data. Die Anpassung erfolgte mit einem Capped Data-Schritt während der Adaption und zehn Schritten abschließend. Die höhere maximale Auswahlordnung der Terme ergab eine insgesamt bessere Anpassung, insbesondere bei der Extrapolation.

FS-AIC generierte stark überanpassende Modelle, FS-BIC solche mit schlechter Verallgemeinerungsfähigkeit (Tabelle 5.15).

q_{max}	$\|D_{test}\|$	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	p_{max}
3	50%	0,97	0,81	0,04	224	1539
4	50%	0,98	0,96	1	107	8854

Tabelle 5.14: *Anpassungsgüte des adaptiven Polynommodells an die Benchmark-Funktion 7 (Cluster-Daten) mit Capped Data-Iterationen.*

	FS-AIC				FS-BIC			
q_{max}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{modell}
3	0,99	0,34	0	794	0,95	0,75	0,04	87
4	0,99	0,08	0,01	794	0,98	0,15	0,02	148

Tabelle 5.15: *Anpassungsgüte der Forward Stepwise Regression an die Benchmark-Funktion 7 mit Capped Data-Iterationen.*

5.1.2 Wasserstoffinduzierte Rissbildung (HIC)

Für das HIC-Problem (Abschnitt 2.1.2) wurde die Wirkung verschiedener Lernparameter des Algorithmus auf die Anpassungsgüte untersucht, die im Folgenden diskutiert werden. Da die Datenmenge zu klein ist, um separate Validierungs-Datensätze zu bilden, wurde zur Bewertung des Modells das Bestimmtheitsmaß R^2_{lern} über alle Lerndaten (Training $\cup$ Test) ermittelt. Zur Anpasung der Termstrukturen wurde der Eingangsdatensatz in einen Trainingsdatensatz zur Bestimmung der Koeffizienten und einen Testdatensatz zur Bewertung aufgeteilt. Die Verallgemeinerungsfähigkeit von Trainings- auf Testdatensatz wurde anhand der über die Kreuzvalidierungsschritte gemittelten MSE_{train} und MSE_{test} verglichen. Je stärker MSE_{test} von MSE_{train} abweicht, desto schlechter verallgemeinert das Modell, so dass für ein gut verallgemeinerndes Modell $MSE_{test}/MSE_{train} \approx 1$ gelten muss. Ein allgemeingültiges Kriterium, wie groß MSE_{test}/MSE_{train} höchstens sein darf, lässt sich aber aus eigener Erfahrung nicht angeben.

In den folgenden Beispielen werden die Effekte der Lernparameter untersucht. Lernparameter sind z. B. die maximale Anpassungsordnung oder die Größe der Test- und Trainingsdatenmenge. Hierbei gilt es u. a., einen vertretbaren Kompromiss aus Rechenzeit und Qualität der Anpassung zu finden. In den folgenden Beispielen wurden zu jedem Satz von Lernparametern mehrere Modelle erstellt, um die Verallgemeinerung der Anpassungsergebnisse zu überprüfen. Die aufgeführten Ergebnisse stammen jeweils von einem Modell, dessen Anpassungsgüte repräsentativ in der Mitte aller Modellergebnisse liegt.

Maximale Ordnung

Im folgenden Beispiel (Abbildung 5.2) wurden die Messdaten zu gleichen Teilen auf die Trainings- und Testdaten verteilt. Die Termstruktur wurde mit der schrittweisen Iterationsvorschrift zum Hinzufügen von $\delta^+ = 2$ und Entfernen von $\delta^- = 1$ Termen angepasst. Es wurde zur Bewertung der Auswahlterme während der Anpassung jeweils ein Capped Data-Schritt durchgeführt.

Tabelle 5.16 zeigt die Modellergebnisse für die maximalen Ordnungen $q_{max} = 2$, $q_{max} = 3$ und $q_{max} = 4$. Bestimmtheitsmaß sowie Trainings- und Testfehler verbessern sich bei Erhöhung der maximalen Ordnung der Auswahlterme.

Ein vollständiges Polynom 3. Grades würde mit 1539 Termen auf den 1578 Messdaten überanpassen. Ein vollständiges Polynom 2. Grades liefert mit $R^2_{lern} = 0,48$ auf allen Lerndaten eine vergleichbare Anpassung wie das Modell mit $q_{max} = 3$, aber schlechter als $q_{max} = 4$. Der Fehler auf den Testdaten $MSE_{test} = 0,434$ war größer als bei allen adaptiven Polynomen und auch deutlich größer als $MSE_{train} = 0,12$, was auf eine schlechte Verallgemeinerungsfähigkeit bzw. starke Überanpassung schließen lässt.

q_{max}	R^2_{lern}	MSE_{train}	MSE_{test}	p_{modell}	p_{max}
2	0,37	0,166	0,173	43	209
3	0,5	0,137	0,146	75	1539
4	0,58	0,115	0,127	85	8854

Tabelle 5.16: *Anpassungswerte an den HIC-Datensatz für verschiedene maximale Ordnungen q_{max}.*

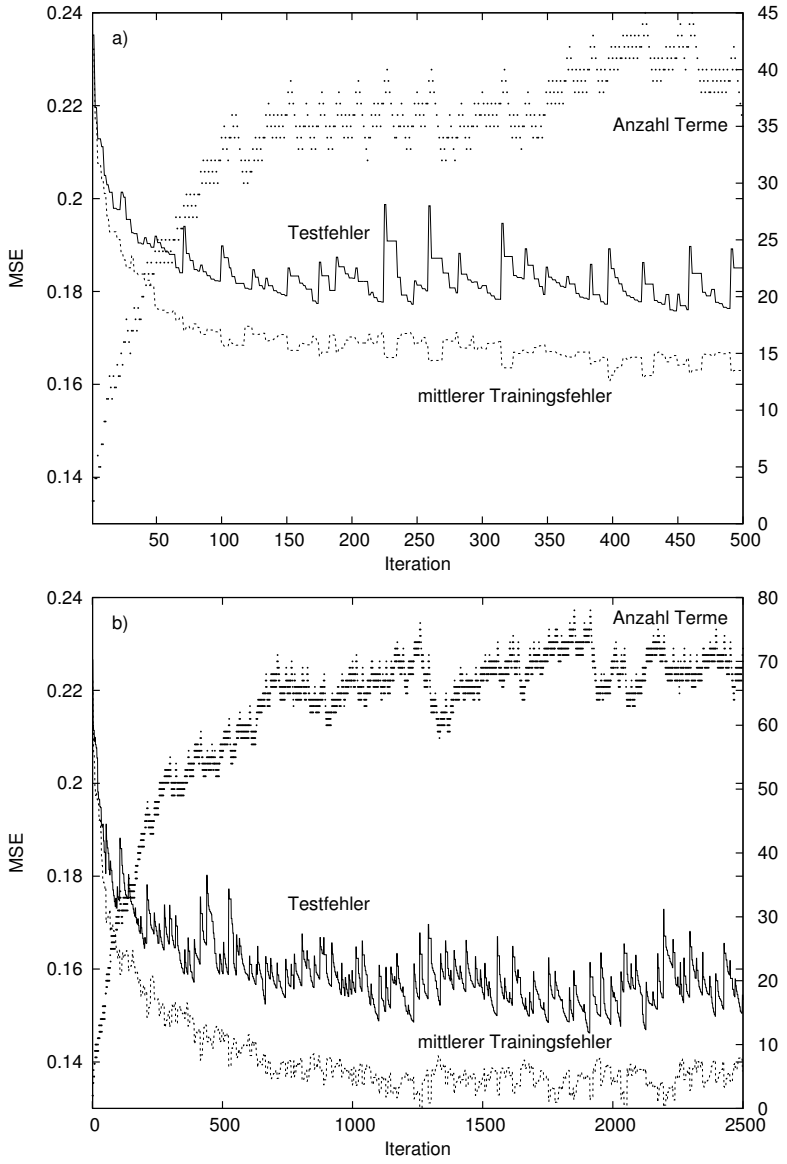


Abbildung 5.2: *Iterationsverläufe für verschiedene maximale Ordnungen der Auswahlterme: Im Fall a) betrug die maximale Ordnung 2, bei Iteration b) 3. Im Fall a) konvergiert die Optimierung der Termstruktur bereits nach etwa 200 Iterationen, im Fall b) erst nach etwa 1000.*

Capped Data

Um die Anpassung nicht durch die Eigenschaft $CAR \geq 0$ einzuschränken, wurden Iterationen mit mehreren Capped Data-Schritten durchgeführt. Ein einzelner Capped Data-Schritt bedeutet dabei, dass bei der Fehlerbewertung lediglich alle negativen Modellwerte als 0 gewertet wurden. *n* Capped Data-Schritte bedeuten, dass danach *n - 1* weitere Modellanpassungen mit durch negative Modellwerte ersetzte Null-

Messwerte durchgeführt wurden.

Im folgenden Beispiel (Tabelle 5.17) wurde die Termstruktur mit der schrittweisen Iterationsvorschrift $\delta^+ = 2, \delta^- = 1$ angepasst bei maximaler Ordnung $q_{max} = 3$ und variabler Anzahl an Capped Data-Schritten n_{cap}. Die endgültige Termstruktur wurde mit zehn Capped Data-Schritten an alle Lerndaten angepasst.

n_{cap}	R^2_{lern}	MSE_{train}	MSE_{test}	p_{modell}
0	0,43	0,146	0,156	64
1	0,497	0,137	0,146	75
2	0,5	0,134	0,143	63

Tabelle 5.17: *Anpassungswerte an den HIC-Datensatz für verschiedene Anzahlen n_{cap} an Capped Data-Schritten während der Entwicklung der Termstruktur.*

Die Korrelationsplots (Abbildung 5.3) zeigen, wie das Modell durch die Capped Data-Schritte positive Werte besser anpassen kann, da die Anpassung nicht durch die Fehlerminimierung bezüglich der $CAR = 0$-Messwerte eingeschränkt wird.

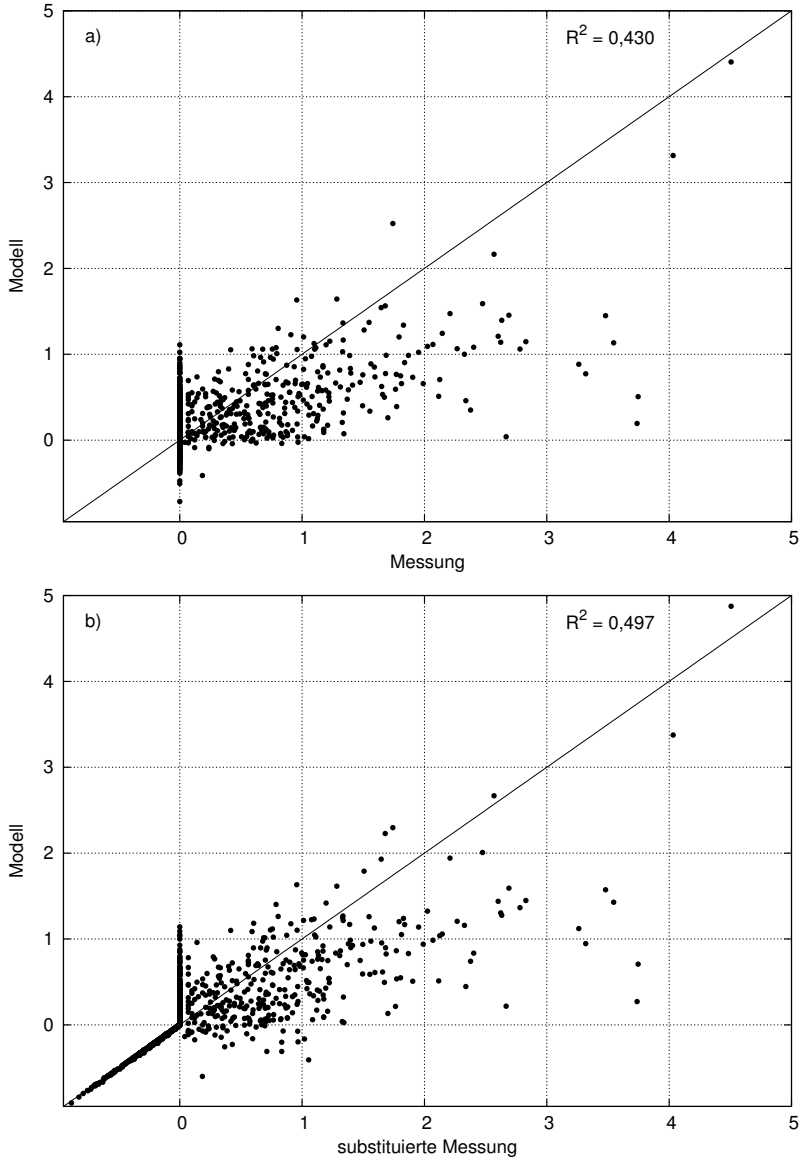


Abbildung 5.3: *Capped Data: Korrelationsplot Messdaten gegen Modelldaten. Iteration a) wurde ohne Capped Data-Schritte durchgeführt, Iteration b) mit einem Capped Data-Schritt. Messwerte mit Wert Null wurden abschließend in zehn Capped Data-Schritten durch negative Modellwerte ersetzt. Durch die Capped Data-Ersetzung wird das Modell in der Anpassung an die positiven Werte weniger stark eingeschränkt, so dass die Modellvorhersagen näher an den Messdaten liegen. Zu beachten ist, dass* R^2 *bezüglich* $\max\{0; Fit(x)\}$ *berechnet wurde.*

Aufteilung der Trainings- und Testdatenmenge

Je größer die Trainingsdatenmenge gewählt wird, desto mehr Detailinformationen stehen zur Anpassung des Datensatzes zur Verfügung. Gleichzeitig nimmt die Verifizierbarkeit durch die ausdünnende Testda-

tenmenge aber immer weiter ab und wird stärker durch Fehlerstreuungen gestört. Eine Vergrößerung der Trainingsdatenmenge erhöht den Rechenaufwand der *QR*-Zerlegungen linear mit der Anzahl der Messungen (Abschnitt 3.7). Die Extrapolation auf die Testdaten ist nach einmaliger Aufstellung der Matrix *A* lediglich eine Matrix-Vektor-Multiplikation. Bei Mehrfach-Kreuzvalidierung ist damit der Rechenaufwand bei großer Trainings- und kleiner Testdatenmenge höher als im umgekehrten Fall.

Im folgenden Beispiel (Tabelle 5.18) wurde die Termstruktur mit der schrittweisen Iterationsvorschrift $\delta^+ = 2$, $\delta^- = 1$ und maximaler Ordnung $q_{max} = 3$ angepasst. Dabei wurden während jedes Iterationsschritts 1 und zum Abschluss des Modells zehn Capped Data-Schritte durchgeführt. Bei einem hohen Anteil an Trainingsdaten und niedrigem Anteil Testdaten kann das Modell insgesamt besser anpassen. Dazu darf der Testfehler aber nicht stark rauschen, was durch die Mittelung über alle Kreuzvalidierungsschritte erreicht wird.

$\lvert D_{test}\rvert$	R^2_{lern}	MSE_{train}	MSE_{test}	p_{modell}
10%	0,68	0,097	0,114	203
25%	0,57	0,121	0,135	117
50%	0,5	0,137	0,146	75

Tabelle 5.18: *Anpassungswerte an den HIC-Datensatz für verschiedene Aufteilungen von Trainings- und Testdaten.*

Anzahl Erweiterungs- und Vereinfachungsschritte

Durch Erhöhung der Anzahl der pro Iterationsschritt hinzugefügten Terme δ^+ (Abschnitt 3.5) erreicht das Polynommodell schneller eine hohe Zahl an Parametern, mit der es komplexe funktionale Zusammenhänge des Datensatzes besser abbilden kann. Dabei durchläuft es aber weniger Vereinfachungsschritte, so dass die Wahrscheinlichkeit höher ist, ungeeignete Termkombinationen beizubehalten.

Im folgenden Beispiel wurde die maximale Ordnung $q_{max} = 3$, eine Aufteilung in 50% Trainings- und 50% Testdaten, sowie je ein Capped Data-Schritt während der Iterationsschritte und zehn zum Abschluss des Modells verwendet.
Bei $\delta^+ = 2$ betrug $R^2_{lern} = 0,5$, $MSE_{test} = 0,146$, $MSE_{train} = 0,137$ und es wurden 75 Terme verwendet.
Bei $\delta^+ = 3$ betrug $R^2_{lern} = 0,54$, $MSE_{test} = 0,173$, $MSE_{train} = 0,166$ und es wurden 105 Terme verwendet.
Abbildung 5.4 zeigt, wie das Modell mit $\delta^+ = 2$ gegen Ende der Iteration ein besseres Verhältnis von MSE_{test} zur Anzahl Terme findet als das Modell mit $\delta^+ = 3$.

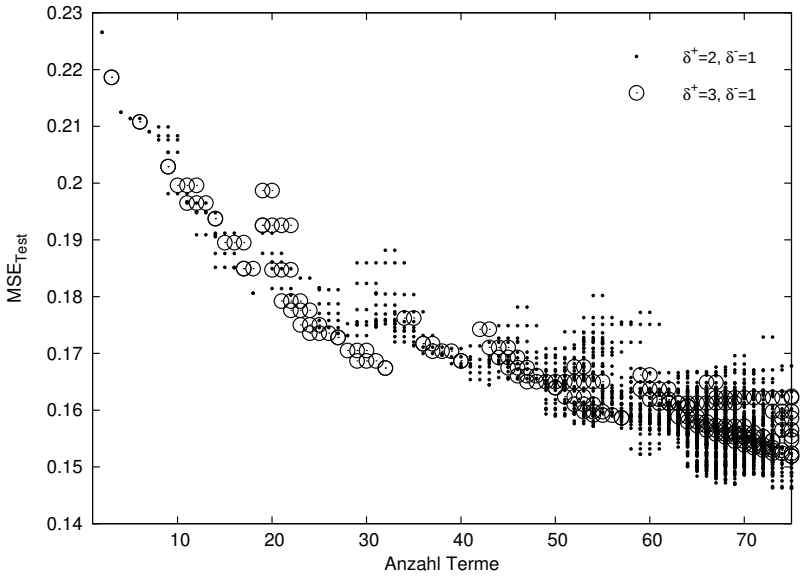


Abbildung 5.4: *Anzahl Erweiterungs- und Vereinfachungsschritte: Zu gleicher Anzahl Terme findet die Iteration mit* $\delta^+ = 2$, $\delta^- = 1$ *einen geringerem Fehler als die Iteration mit* $\delta^+ = 3$, $\delta^- = 1$.

Generalisierungsfähigkeit

Der Vergleich des mittleren Test- und Trainingsfehlers während der Iteration liefert bereits ein Maß für die Generalisierungsfähigkeit des Modells. Da die Trainingsdaten aber insofern Einfluss auf das Modell haben, als sie den Extrapolationsfehler zur Bewertung der Termstruktur bestimmen, wurde der Messdatensatz in der folgenden Versuchsreihe um einen zufällig ausgewählten Validierungsdatensatz reduziert, für den das abgeschlossene Modell eine Vorhersage machen sollte. Für die Anpassung der Termstruktur wurde eine Iteration mit maximaler Ordnung $q_{max} = 3$ und $\delta^+ = 2$, $\delta^- = 1$ gewählt. Dabei wurden während der Entwicklung jeweils ein und zum Abschluss zehn Capped Data-Schritte durchgeführt. Der Validierungsdatensatz bestand aus 15% der Messdaten. Der Lerndatensatz aus den restlichen 85% wurde zufällig in Trainings- und Testdaten aufgeteilt.

Bei Modell a) wurden die Daten zu 50% auf die Trainings- und zu 50% auf die Testmenge verteilt. Das Bestimmtheitsmaß auf den Lerndaten betrug $R^2_{lern} = 0,50$ und auf den Validierungsdaten $R^2_{valid} = 0,43$. Um die Generalisierungsfähigkeit weiter zu verbessern, wurde versucht, die Streuung des Modells durch eine Mittelung der Ergebnisse aus insgesamt vier Einzelmodellen zu verringern. Dabei ergab sich eine geringfügige Verbesserung von $R^2_{lern} = 0,53$ und $R^2_{valid} = 0,45$.

Bei Modell b) wurden die Daten zu 90% auf die Trainings- und zu 10% auf die Testmenge verteilt. Das Bestimmtheitsmaß auf den Lerndaten betrug $R^2_{lern} = 0,65$ und auf den Validierungsdaten $R^2_{valid} = 0,63$. Durch Mittelung über vier Einzelmodelle verbesserte sich das Ergebnis zu $R^2_{lern} = 0,66$ und $R^2_{valid} =$

$0,67$.

Zum Vergleich wurden mit dem selben Trainings- und Validierungsdatensatz vollständige Polynome mit zehn Capped Data-Schritten angepasst. Eine lineare Anpassung ergab $R^2_{valid} = 0,15$ und eine quadratische Anpassung ergab $R^2_{valid} = 0,14$. Die Datenmenge ließ gerade noch eine vollständige Anpassung des Polynoms

$$f_{voll}(x) = const + \prod_{i=1}^{n} a_i x_i + \prod_{i,j=1}^{n} a_{ij} x_i x_j + \prod_{\substack{i,j,k=1 \\ i \neq j \neq k}}^{n} a_{ijk} x_i x_j x_k \tag{5.1.1}$$

mit 1178 Termen zu. Daraus resultierte eine starke Überanpassung mit $R^2_{lern} = 0,95$ und $R^2_{valid} = 0,002$.

Vergleich mit Forward Selection

Der in Trainings- und Validierungsdaten aufgeteilte Datensatz wurde zur Bewertung der Stepwise Forward Regression mit AIC- bzw. BIC-Abbruchkriterium mit $p_{max} = 3$ und ein CAP-Schritt verwendet. Ähnlich der 19-dimensionalen Benchmark-Funktion 7 (Abschnitt 5.1.1) generierte FS-AIC ein stark überanpassendes und FS-BIC ein zu starres Modell mit schlechter Generalisierungsfähigkeit (Tabelle 5.19).

FS-AIC			FS-BIC		
R^2_{lern}	R^2_{valid}	p_{modell}	R^2_{lern}	R^2_{valid}	p_{modell}
0,86	0	672	0,38	0,18	27

Tabelle 5.19: *Anpassungsgüte der Forward Stepwise Regression an die Benchmark-Funktion 4 mit Capped Data-Iterationen.*

Vergleich mit Backward Elimination und Ridge Regression

Der Test- und Validierungsdatensatz aus Abschnitt 5.1.2 wurde mit einer Stepwise Regression mit Backward Elimination modelliert, die in der Data Mining-Software KNIME [112] unter Einbindng der WEKA-Bibliothek[113] implementiert ist. Dieser Algorithmus führt eine Ridge Regression aus (Gleichung 3.1.9), zu der ein Ridge-Parameter λ vorgegeben werden muss. Anschließend werden nicht signifikante Terme schrittweise mittels eiens t-Tests eliminiert und das Folgemodell jeweils mit einer Ridge Regression angepasst. Durch die Ridge Regression kann die Backward Elimination mit dem vollständigen Modell beginnen, obwohl es unterbestimmt ist.

λ	R^2_{lern}	R^2_{valid}	MSE_{lern}	MSE_{valid}	p_{modell}
10^{-1}	0,00	0,00	0,280	0,154	0
$5 \cdot 10^{-2}$	0,33	0,00	0,188	4,970	37
$2,5 \cdot 10^{-2}$	0,51	0,022	0,139	0,851	132
10^{-2}	0,53	0,08	0,133	0,151	150
10^{-3}	0,55	0,00	0,129	7,104	171
10^{-4}	0,56	0,02	0,127	4,978	181
10^{-5}	0,51	0,01	0,139	75,592	148
10^{-6}	0,57	0,00	0,124	18,299	192
10^{-7}	0,59	0,00	0,118	0,167	210
10^{-8}	0,58	0,00	0,119	22,372	206
0 (Least Squares)	0,55	0,00	0,129	2,031	161

Tabelle 5.20: *Anpassungswerte an den HIC-Datensatz mit Stepwise Ridge Regression und Backward Elimination.*

Das Verfahren generiert im Vergleich zum hier vorgestellten adaptiven Polynommodell im Schnitt eine etwas höhere Anzahl Terme, wobei diese bei beiden Algorithmen von der Wahl der Parameter abhängt. Für eine hohe Anzahl an Kreuzvalidierungsschritten generiert das adaptive Polynommodell eine vergleichbare Anzahl Terme (Siehe Abschnitt 5.1.2). Die Wiedergabe der Lerndaten erfolgt mit der Backward Elimination geringfügig besser als im adaptiven Polynommodell, die Vorhersage auf die Validierungsdaten ist dagegen deutlich schlechter.

5.1.3 Zugfestigkeit

Der Datensatz der Zugfestigkeit (Abschnitt 2.1.1) wurde mit maximaler Ordnung $q_{max} = 3$, dem Termentwicklungsschema $\delta^+ = 2$, $\delta^- = 1$ und einer Aufteilung der Messdaten auf 33% Test- und 67% Trainingsdaten angepasst (Abbildung 5.5). Eine vorherige Transformation der Messdaten oder Capped Data-Schritte waren aufgrund der gleichmäßigen Verteilung und des Wertebereichs nicht nötig. Die Anpassung ergab eine Termstruktur aus $p_{modell} = 205$ von $p_{max} = 1539$ Termen und $R^2_{lern} = 0,91$. Der Fehler auf den Trainingsdaten $MSE_{test} = 125,2$ ist lediglich un 3% größer als der $MSE_{train} = 121,1$, was eine gute Verallgemeinerungsfähigkeit im Bereich der Lerndaten impliziert. Bei größeren Trainingsdatenmengen und entsprechend hoher Zahl an Kreuzvalidierungen ist die Erstellung der Modelle sehr rechenintensiv.

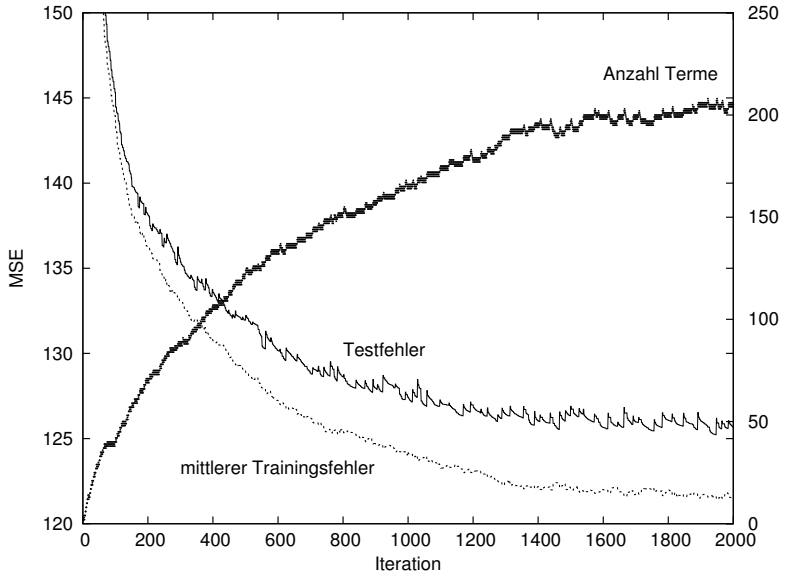


Abbildung 5.5: *Iterationsverlauf der Anpassung an die Zugfestigkeits-Daten.*

Als weiterer Validierungstest des Algorithmus wurde aus allen Messdaten eine zufällige Teilmenge von insgesamt 2000 Daten zum Lernen entnommen und zum Erstellen von Polynommodellen in Trainings- und Testdaten aufgeteilt. Die Ergebnisse mit Validierung auf den restlichen 24.203 Daten sind in Tabelle 5.21 zusammengefasst. Tabelle 5.22 zeigt den Verlauf für Forward Selection mit AIC und BIC.

Kleine Testdatensätze und entsprechend hohe Anzahlen an Kreuzvalidierungen lieferten bei diesem Datensatz die am besten anpassenden Modelle. R^2_{lern} nahm gegenüber der Anpassung an alle Daten zu, da ein kleinerer Datensatz angepasst wurde und somit das Verhältnis der Anzahl Modellterme zu Trainingsdaten in jedem Schritt größer war. Gleichzeitig nahm die Generalisierungsfähigkeit ab, was am größeren Unterschied zwischen MSE_{train} und MSE_{test} von 9-11% im Vergleich zu 3% beim vollen Datensatz ersichtlich wird. Die direkte Validierung durch R^2_{valid} bestätigt eine gute Verallgemeinerungsfähigkeit.

FS-BIC erzeugte ein Modell mit vergleichbar guter Anpassungs- und Generalisierungsfähigkeit. Das mit FS-AIC generierte komplexe Modell zeigte eine starke Überanpassung (Tabelle 5.22).

$\lvert D_{test}\rvert$	R^2_{lern}	R^2_{valid}	MSE_{train}	MSE_{test}	p_{modell}
10%	0,94	0,85	95,1	105,9	124
25%	0,94	0,85	95,9	107,2	110
33%	0,94	0,85	97	107,8	106
50%	0,93	0,86	106,2	115,3	72

Tabelle 5.21: *Anpassungswerte an den Zugfestigkeits-Datensatz für verschiedene Aufteilungen von Trainings- und Testdaten.*

FS-AIC			FS-BIC		
R^2_{lern}	R^2_{valid}	p_{modell}	R^2_{lern}	R^2_{valid}	p_{modell}
0,978	0,05	1000	0,93	0,87	56

Tabelle 5.22: *Anpassungswerte der Forward Regression mit AIC und BIC an den Zugfestigkeits-Datensatz.*

5.1.4 Oszillationsmarken

Für den Datensatz der Oszillationsmarken (Abschnitt 2.1.3) hat sich die maximale Ordnung 4 als geeignet erwiesen (Abbildung 3.4 auf Seite 52). Durch eine Transformation der Zielgröße $d_{OSM} \rightarrow \sqrt{d_{OSM}}$ wurde eine geringfügige Verbesserung der Anpassungsgüte der rücktransformierten Zielgröße im Vergleich zu Anpassungen an die untransformierte Zielgröße erreicht. Wegen der geringen Datenmenge fand der Algorithmus mit 75% Trainings- und 25% Testdaten bereits nach etwa zehn Iterationsschritten eine Termstruktur, die sich im weiteren Verlauf nur noch sehr langsam weiter verbessern ließ. Tabelle 5.23 fasst die Ergebnisse zusammen:

adaptives Polynom		FS-AIC		FS-BIC	
R^2_{train}	p_{modell}	R^2_{train}	p_{modell}	R^2_{train}	p_{modell}
0,56	15	0,58	34	0,54	8

Tabelle 5.23: *Anpassungswerte der Forward Regression mit AIC und BIC an den Oszillationsmarken-Datensatz.*

Die Daten konnten von allen Algorithmen nur mit mäßiger Genauigkeit wiedergegeben werden. Wegen der geringen Menge an Daten wurde auf eine Abspaltung eines Validierungsdatensatzes verzichtet. Beim adaptiven Polynom ergab sich während der Kreuzvalidierung ein mittleres $R^2_{test} = 0,54$, so dass keine deutliche Tendenz zur Überanpassung ersichtlich ist.

5.1.5 Rechenzeit

Von den untersuchten Verfahren gelangte die Forward Selection mit BIC fast immer am schnellsten zum finalen Modell, da hier meistens weniger Terme als bei der Forward Selection mit AIC oder dem adaptiven Polynomalgorithmus aus dieser Arbeit verwendet wurden. Die FS-AIC war trotz der in den meisten Fällen deutlich höheren Zahl an verwendeten Termen schneller als der adaptive Polynomalgorithmus, da dieser aufgrund der Vereinfachungsschritte mehr Iterationsschritte benötigte, um eine gegebene Anzahl an Termen zu erreichen. Nur bei hochdimensionalen Problemen ($m > 10$) und vielen Messdaten überwog letztendlich die Rechendauer der komplexen Kandidatenmodelle mit hohem p. Ebenso war die Backward

Elimination deutlich langsamer als der adaptive Polynomalgorithmus, da alle finalen Modelle eine deutlich geringere Komplexität hatten als das vollständige, $p_{modell} \ll p_{max}$ und daher eine hohe Anzahl an Berechnungen sehr komplexer Modelle durchgeführt werden musste.

5.2 Neuronales Netz mit adaptiver Topologie

5.2.1 Anpassungsoptionen

Die Bewertung der Topologie der Kandidatennetze erfolgte über den minimalen Fehler auf den Testdaten MSE_{test} über alle Lernepochen. Der Lernprozess wurde so lange fortgesetzt, bis ein lokales Minimum gefunden wurde oder eine maximale Anzahl Epochen erreicht wurde.

Um dem Lernprozess genügend Zeit zur Konvergenz zu geben, wurde jedes Training zunächst mindestens s_{min} Schritte lang durchgeführt. Um ein zu frühzeitiges Abbrechen aufgrund von Schwankungen des Testfehlers zu vermeiden, wurde nicht sofort bei Zunahme des Testfehlers vom Schritt i nach $i+1$ abgebrochen, sondern mindestens um einen vorgegebenen Faktor der bisherigen Anzahl Schritte weiter trainiert. Eine angemessene Anzahl ist vom Problem abhängig. Bei schnell abnehmendem Fehler braucht nicht so lange trainiert zu werden wie bei langsam abnehmendem Fehler. Der Trainingsschritt s^i_{best}, in dem der bisher beste Testfehler gefunden wurde, wurde hierbei als ein Maß für die Konvergenzgeschwindigkeit betrachtet. Das Training wurde dann noch $n_{min} \cdot s^i_{best}$ Schritte fortgeführt, um nach einem noch günstigeren Fehlerwert zu suchen. Der Faktor n_{min} ist vom Benutzer vorgegeben. In den folgenden Fällen wurde $n_{min} = 2$ gewählt. Sollte innerhalb dieser Epochen keine weitere Verbesserung stattgefunden haben, so wurde die Annahme gemacht, dass ein Minimum gefunden wurde. Andernfalls wurde das Training um mindestens weitere $n_{min} \cdot s^i_{best}$ Schritte fortgesetzt.

Damit das Training eines einzelnen Netzes nicht zu zeitaufwändig wurde, wurde zusätzlich eine maximale Anzahl s_{max} an Trainingsschritten vorgegeben. Falls in dieser Zeit kein lokales Minimum des Testfehlers durchlaufen wurde, wurde stellvertretend der Testfehler des letzten Iterationsschritts verwendet. Das kann passieren, wenn die Topologie keine hinreichend komplexe Anpassung zulässt, also unteranpasst, wobei der Testfehler gegen ein Minimum konvergiert. Als Abschätzung für eine obere Schranke von s_{max} wurde die Anzahl Iterationen verwendet, die ein vorab probeweise trainiertes vollständiges MLP zum Finden eines minimalen Testfehlers benötigte.

Das Kandidatennetz mit dem geringsten Testfehler wurde als das beste betrachtet und diente als Ausgangsnetz für die folgende Topologieiteration.

Da bei der schrittweisen Entwicklung der Netztopologie viele Kandidatennetze miteinander verglichen werden müssen, sind effiziente Trainingsalgorithmen nötig, die möglichst repräsentative Bewertungen der einzelnen Topologien zulassen. Idealerweise sollten die Kandidatennetze mithilfe des Testfehlers

am globalen Minimum bewertet werden. Da es aber beim Lernen keine Garantie gibt, dieses auch zu finden, müssen statt dessen die Fehler lokaler Minima zum Vergleich herhalten. Insbesondere bei umfangreichen Netztopologien und den damit verbundenen Fehlerlandschaften mit vielen lokalen Mimima kann die Bewertung durch nicht repräsentative Fehlerwerte ungünstiger lokaler Minima gestört werden. Daher ist es sinnvoll, die Fehlerfunktion großräumig abzusuchen. Der SARprop-Algorithmus (Abschnitt 7.2.4.5) eignet sich, um anfängliche lokale Minima zugunsten besserer Lösungen zu verlassen. Bei den Auswertungen konnte dieser Algorithmus zu einer vorgegebenen Anzahl Iterationen in den meisten Fällen die beste Lösung im Vergleich zu den anderen Algorithmen (Abschnitt 7.2.3) finden und wurde deshalb im Folgenden ausschließlich angewendet. Um die Wahrscheinlichkeit, das globale Minimum oder zumindest ein gutes lokales Minimum zu finden, weiter zu erhöhen, wurden für jede Netztopologie mehrere Instanzen mit unterschiedlichen zufälligen Initialisierungen der Gewichte parallel trainiert. Die Instanz mit minimalem Testfehler bestimmt dann die weiter zu verwendende Topologie. Von einer Vererbung der Gewichte während der Topologieänderungen wurde abgesehen.

Die Neuronen in den verdeckten Ebenen wurden durch eine tanh-Funktion aktiviert und das Ausgabeneuron linear. Von einer zufälligen Neuverteilung der Daten auf den Trainings- und Testdatensatz, wie bereits bei den Polynommodellen (Abschnitt 3.8) beschrieben, wurde abgesehen. Dies hätte zusätzliche Schwankungen in die Bewertung der einzelnen Kandidatennetze zur Folge, die ohnehin schon aufgrund des Vergleichs lokaler Minima erschwert wird.

5.2.2 Auswertung

Die Daten zum Lernen wurden in 50% Trainings- und 50% Testdaten aufgeteilt. Nach Abschluss der Topologieoptimierung wurden zehn separate Instanzen des Modells neu trainiert, um die Fehlerlandschaft großflächiger abzusuchen und somit eine möglichst gute Gewichtsoptimierung zu finden. Die Instanz mit dem besten Fehler auf den Testdaten ist in den Tabellen dargestellt. Das Bestimmtheitsmaß des Trainingsfehlers bezieht sich auf die Iteration, bei der der beste Testfehler gefunden wurde. Zum Vergleich wurden ebenfalls zehn Instanzen der vollständigen Netztopologie mit den gleichen Lernparametern trainiert und die beste ausgewählt. Da alle Verbindungen des adaptiven Modells in der vollständigen Topologie enthalten sind, ist zu erwarten, dass die vollständige Topologie bei hinreichend ausführlichem Training ein Ergebnis liefert, das mindestens so gut ist wie das des adaptiven Modells. Wegen der Komplexität der Fehlerlandschaft ist ein solches Training aber unpraktikabel.

5.2.3 Benchmark-Funktionen

Benchmark-Funktion 1 (2D)

Tabelle 5.24 zeigt die Anpassungsstatistiken für neuronale Netze mit vorgegebener maximaler Topologie an die 2-dimensionale Benchmark-Funktion 1 (Gleichung 2.2.1 auf Seite 29). Die Anpassung des adaptiven Modells mit p_{adapt} Verbindungen und der vollständigen Topologie mit p_{vollst} Verbindungen sind identisch, das adaptive Netz kam lediglich mit weniger Gewichten aus. Die 2-2-1-Topologie war nicht komplex genug, um die Funktion mit $R^2_{lern} \approx 1$ wiederzugeben, bei allen anderen Topologien wurde dies erreicht. $R^2_{valid,int}$ bestätigt eine gute Generalisierungsfähigkeit im Bereich der Trainingsdaten. Bei der Extrapolation ist das Bestimmtheitsmaß jedoch deutlich schlechter.

	adaptives Modell			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{adapt}
2-2-1	0,93	0,92	0,48	9
2-4-4-1	1	1	0,42	35
2-10-1	0,99	0,99	0,43	39
2-10-2-1	1	1	0,42	36
	vollständige Topologie			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{vollst}
2-2-1	0,93	0,92	0,48	9
2-4-4-1	1	1	0,42	37
2-10-1	0,99	0,99	0,43	41
2-10-2-1	1	1	0,42	55

Tabelle 5.24: *Anpassungsgüte des adaptiven neuronalen Netzes an die Benchmark-Funktion 1.*

Benchmark-Funktion 2 (10D)

In Tabelle 5.25 ist die Anpassung an die 10-dimensionale Benchmark-Funktion 2 (Gleichung 2.2.2 auf Seite 29) zusammengefasst. Mit steigender Komplexität der maximal vorgegebenen Topologie wurde die Anpassung an die Trainings- und Interpolationsdaten sowohl beim adaptiven Modell als auch beim vollständigen Netz besser. Die Vorhersage für den Interpolationsdatensatz war dabei jedoch schlechter als die Anpassung an die Trainingsdaten. Die Extrapolation fiel insgesamt schlecht aus. Die adaptierten Netze erzielten mit weniger Verbindungen durchgehend bessere Ergebnisse als die vollständigen Netze. Beide Netztypen erzielten für Topologien 10-10-1 und größer gegen Ende der Lerniteration oft $R^2_{train} \approx 1$ mit $R^2_{valid} \approx 0$ für Inter- und Extrapolation. Die Topologien waren daher durchaus komplex genug, um die Trainingsdaten gut anzupassen.

Eine weitere Erhöhung der Komplexität führt neben dem ohnehin höheren Rechenaufwand zu dem

Problem, dass die Ergebnisse einzelner Modelle aufgrund der komplexeren Fehlerlandschaft stärker streuen, so dass noch mehr Instanzen parallel trainiert werden müssen, um repräsentative Ergebnisse zu erzielen.

	adaptives Modell			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{adapt}
10-2-1	0,37	0,34	0,01	13
10-5-1	0,48	0,36	0,01	47
10-10-1	0,69	0,34	0	76
10-20-1	0,85	0,55	0,01	80
	vollständige Topologie			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{vollst}
10-2-1	0,32	0,19	0,03	25
10-5-1	0,38	0,19	0,05	61
10-10-1	0,54	0,26	0	121
10-20-1	0,52	0,26	0,17	241

Tabelle 5.25: *Anpassungsgüte des adaptiven neuronalen Netzes an die Benchmark-Funktion 2.*

Abbildung 5.6 zeigt den Verlauf von Trainings- und Testfehler bei der Anpassung der Netzstruktur an eine 10-20-1-Topologie. Das Ausgangsnetz war nicht komplex genug, um die Daten präzise wiederzugeben, daher verbesserten sich beim Hinzufügen von Verbindungen zunächst sowohl Testfehler als auch Trainingsfehler. Nach 63 Iterationen wurde ein Minimum des Testfehlers erreicht. Da das Lernen des Netzes den Trainingsfehler optimiert, konnte danach keine weitere Verbesserung des Testfehlers garantiert werden und die folgenden Netzstrukturen begannen überanzupassen: $MSE_{test} \gg MSE_{train}$.

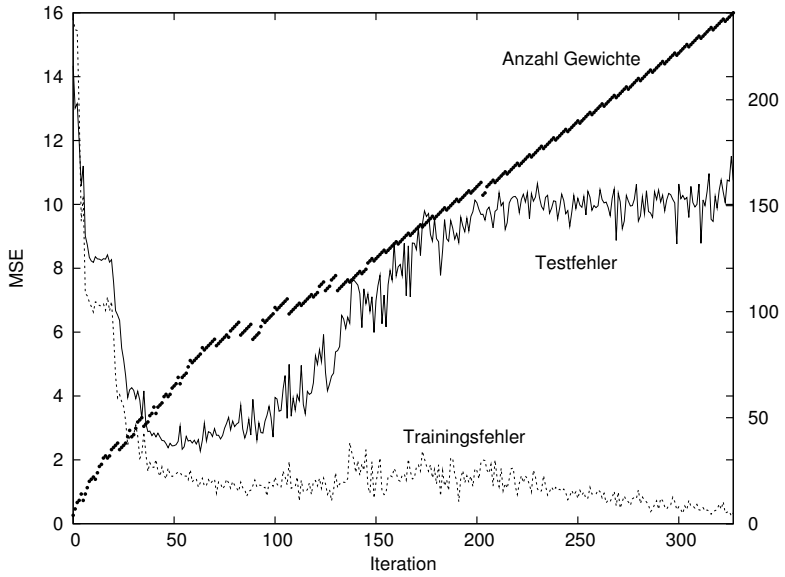


Abbildung 5.6: *Iterationsverlauf der Anpassung der Netzstruktur für eine 10-20-1 Topologie.*

Abbildung 5.7 zeigt die ersten 250 Iterationen des Lernens a) der am besten adaptierten Netztopologie sowie b) der vollständigen Topologie. Die Streuung des Fehlers während der ersten 100 Iterationen ist die Folge des statistischen Absuchens der Fehlerfunktion nach dem Simulated Annealing-Ansatz beim SARprop-Algorithmus. Mit wachsender Anzahl Lernepochen ging der Algorithmus allmählich in einen Gradientenabstiegs-Algorithmus über und der Fehler konvergierte gegen ein Minimun. Beim Training der optimal adaptierten Topologie nahm der Testfehler beim Optimieren des Lernfehlers nicht mehr zu. Das bedeutet, dass diese Topologie nicht überangepasst hat. Im Gegensatz dazu durchlief der Testfehler beim Lernen der vollständigen Topologie ein deutliches Minimum. Hierbei wurde der Testfehler durch die frühe Überanpassung auf die Lerndaten verschlechtert und gleichzeitig fiel der Lernfehler aufgrund des early stopping (Abschnitt 7.2.4.6) ebenfalls schlechter aus als im Fall der adaptierten Topologie.

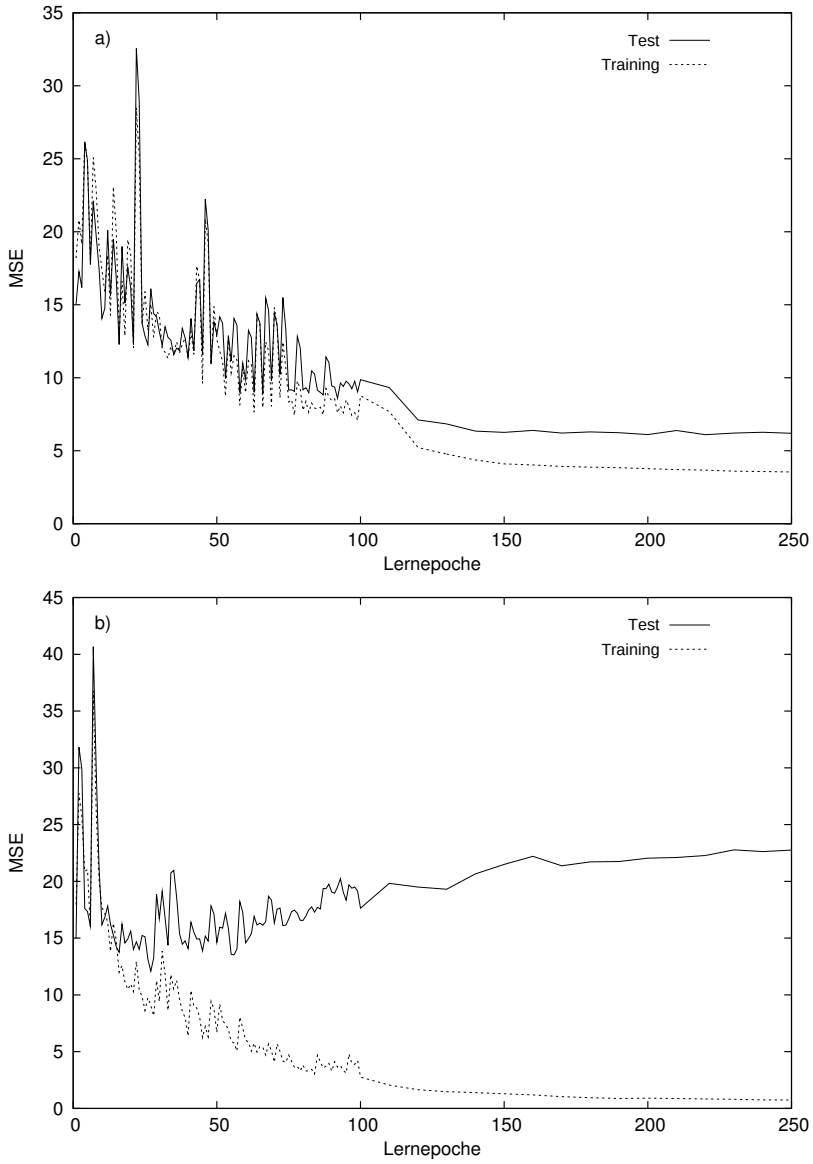


Abbildung 5.7: *Verlauf des Fehlers während des Lernens: a) aus dem Verlauf der Topologieanpassung (Abbildung 5.6) ausgewähltes Kandidatennetz mit optimal angepasster Netzstruktur, b) Netz mit vollständiger 10-20-1-Topologie.*

Benchmark-Funktion 3 (10D, verrauscht)

Tabelle 5.26 zeigt die Anpassung an die 10-dimensionale Benchmark-Funktion 3 mit überlagertem Rauschen (Gleichung 2.2.3 auf Seite 29). Die Ergebnisse zeigen die gleiche Tendenz wie die Anpassung an die nicht verrauschte Funktion.

	adaptives Modell			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{adapt}
10-10-1	0,73	0,48	0	40
10-20-1	0,87	0,54	0	54
	vollständige Topologie			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{vollst}
10-10-1	0,59	0,28	0	121
10-20-1	0,72	0,22	0	241

Tabelle 5.26: *Anpassungsgüte des adaptiven neuronalen Netzes an die verrauschte Benchmark-Funktion 3.*

Benchmark-Funktion 4 (10D, Capped)

Tabelle 5.27 fasst die Ergebnisse der Anpassung an die Benchmark-Funktion 4 mit Capped Data (Gleichung 2.2.4 auf Seite 29) zusammen. Beim Training der Netze wurde die Capped Data-Eigenschaft, nach der alle negativen Funktionswerte Null gesetzt wurden, dadurch berücksichtigt, dass der Bewertungsfehler der Ausgabeebene des Netzes bezüglich $\max(f_w(x_1,\ldots,x_m);0)$ berechnet wurde. Auf den Trainingsdaten erzielte das adaptive Modell gute Ergebnisse. Die Interpolations-Validierungsdaten waren allerdings schlechter und bei der Extrapolation versagte das Modell. Die Anpassung der vollständigen Topologie lieferte durchgehend schlechtere Ergebnisse als das adaptive Modell.

	adaptives Modell			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{adapt}
10-10-1	0,82	0,57	0,06	53
10-20-1	0,91	0,57	0	80
	vollständige Topologie			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{vollst}
10-10-1	0,5	0,24	0	121
10-20-1	0,75	0,16	0,01	241

Tabelle 5.27: *Anpassungsgüte des adaptiven neuronalen Netzes an die Benchmark-Funktion 4 mit Capped Data-Bewertung.*

Benchmark-Funktion 5 (10D, Capped, verrauscht)

In Tabelle 5.28 ist die Anpassung an die 10-dimensionale Benchmark-Funktion 5 mit Capped Data und überlagertem Rauschen (Gleichung 2.2.5 auf Seite 29) dargestellt. Zum Vergleich zeigt Tabelle 5.29 die Anpassung ohne Berücksichtigung der Capped Data-Eigenschaft beim Training. Wenn die Capped Data-Eigenschaft nicht explizit berücksichtigt wird, benötigt ein Netz mehr Verbindungen und Neuronen, um

den anstelle des kontinuierlichen Übergangs entstehenden „Knick" von positiven Werten zu Null-Werten zu modellieren (vergleiche Abbildung 3.9). Diese erhöhte Komplexität erschwert die Anpassung, was sich in den niedrigeren Werten von R^2_{train} widerspiegelt.

	adaptives Modell			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{adapt}
10-10-1	0,8	0,59	0,01	38
10-20-1	0,8	0,53	0,01	53
	vollständige Topologie			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{vollst}
10-10-1	0,54	0,11	0,06	121
10-20-1	0,73	0,11	0	241

Tabelle 5.28: *Anpassungsgüte des adaptiven neuronalen Netzes an die verrauschte Benchmark-Funktion 5 mit Capped Data-Bewertung.*

	adaptives Modell			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{adapt}
10-10-1	0,75	0,37	0,01	38
10-20-1	0,79	0,45	0	50
	vollständige Topologie			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{vollst}
10-10-1	0,54	0,4	0,06	121
10-20-1	0,57	0,24	0,01	241

Tabelle 5.29: *Anpassungsgüte des adaptiven neuronalen Netzes an die verrauschte Benchmark-Funktion 5 ohne Capped Data-Bewertung.*

Benchmark-Funktion 6 (10D, Capped, verrauscht, in Clustern angeordnet)

Durch die Clusterung der Benchmark-Funktion 6 (Gleichung 2.2.6 auf Seite 29) wurde die Generalisierung weiter erschwert. Da dabei aber gegenüber einer Gleichverteilung (5.2.3) viele Daten in einem kleinen Intervall liegen, ist zu erwarten, dass die Spezialisierung auf die Trainingsdaten hier zu besseren Anpassungen führt. Das ist in Tabelle 5.30 dargestellt.

	adaptives Modell			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{adapt}
10-10-1	0,88	0,18	0,12	64
10-20-1	0,79	0,11	0,01	84
	vollständige Topologie			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{vollst}
10-10-1	0,8	0,03	0,04	121
10-20-1	0,65	0,12	0,07	241

Tabelle 5.30: *Anpassungsgüte des adaptiven neuronalen Netzes an die verrauschte, Benchmark-Funktion 6 (Cluster-Daten) mit Capped Data-Bewertung.*

Benchmark-Funktion 7 (10D, Capped, verrauscht, in Clustern angeordnet)

Tabelle 5.31 zeigt die Anpassung an die 19-dimensionale Benchmark-Funktion 7 (Gleichung 2.2.7 auf Seite 30). Die Interpolation liefert bessere Ergebnisse als bei der Modellierung der analogen 10-dimensionale Benchmark-Funktion 6, das Netz ist aber nicht extrapolationsfähig.

	adaptives Modell			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{adapt}
19-19-1	0,89	0,73	0	76
	vollständige Topologie			
Topologie	R^2_{train}	$R^2_{valid,int}$	$R^2_{valid,ext}$	p_{vollst}
19-19-1	0,93	0,59	0	400

Tabelle 5.31: *Anpassungsgüte des adaptiven neuronalen Netzes an die verrauschte, Benchmark-Funktion 6 (Cluster-Daten) mit Capped Data-Bewertung.*

5.2.4 Zugfestigkeit

Um die Topologieanpassung zu beschleunigen, wurde wie beim adaptiven Polynommodell (Abschnitt 5.1.3) aus allen vorhandenen Messdaten ein zufällig gewählter Trainingsdatensatz aus 2000 Messungen verwendet und in Trainings- und Testdaten aufgeteilt. Die restlichen 24.203 Messungen wurden zur Validierung verwendet. Dadurch wird eine sehr strenge Überprüfung der Verallgemeinerungsfähigkeit ermöglicht.

Die Anpassung an eine 19-19-1-Topologie mit dem Netzentwicklungsschema $\delta^+ = 6$, $\delta^- = 1$ ergab ein optimales Netz mit 264 Verbindungen und $R^2_{train} = 0,93$, $R^2_{test} = 0,84$ sowie $R^2_{valid} = 0,8$.

Die vollständige Topologie mit 400 Verbindungen ergab mit $R^2_{train} = 0,93$, $R^2_{test} = 0,85$ und $R^2_{valid} =$

$0,81$ eine marginal bessere Anpassung. Da die optimierte Topologie jedoch lediglich 2/3 der Verbindungen benötigte, kann beim Hinzufügen neuer Daten eine robustere Anpassung erwartet werden.

Abbildung 5.8 zeigt, dass sich der Testfehler nach 50 Iterationen auf dem wenig verrauschten Datensatz nur noch geringfügig ändert, während der Validierungsfehler stark oszilliert. Während der Topologieoptimierung wurden die Kandidatennetze nicht notwendigerweise bis zu einem lokalen Optimum des Testfehlers trainiert, sondern nach einer vorgegebenen maximalen Anzahl an Lernepochen abgebrochen. Erst das finale Netz wurde bis zu einem Optimum trainiert. Die Bestimmtheitsmaße deuten trotz des während der Topologieoptimierung deutlich höheren Validierungsfehlers auf eine akzeptable Verallgemeinerungsfähigkeit der finalen Netze hin.

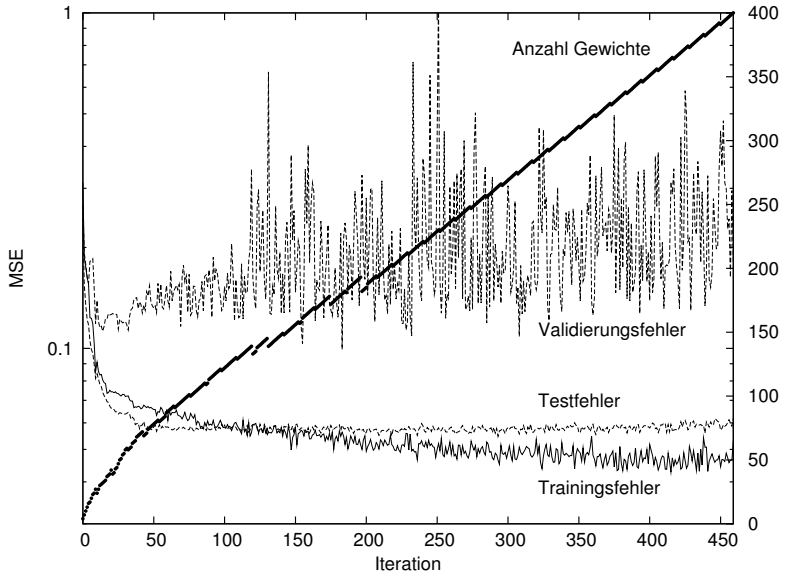


Abbildung 5.8: *Iterationsverlauf der Anpassung der Netztopologie an die Zugfestigkeits-Daten an eine 19-19-1-Topologie.*

5.2.5 Wasserstoffinduzierte Rissbildung (HIC)

Tabelle 5.32 zeigt die Ergebnisse der Anpassung einer 19-19-1- und einer 19-20-2-1-Topologie. Die Messdaten wurden wie in beim adaptivem Polynommodell (Abschnitt 5.1.2) zu 85% auf Lerndaten und 15% Validierungsdaten aufgeteilt. Die Lerndaten wurden in gleichen Teilen auf Trainings- und Testdaten aufgeteilt. Die Topologieanpassung erfolgte unter Berücksichtigung der Capped Data-Eigenschaft.

	adaptives Modell				vollständige Topologie			
Topologie	R^2_{train}	R^2_{test}	R^2_{valid}	p_{adapt}	R^2_{train}	R^2_{test}	R^2_{valid}	p_{vollst}
19-19-1	0,5	0,27	0,13	80	0,4	0,19	0,15	400
19-20-2-1	0,51	0,19	0,09	53	0,45	0,21	0,13	445

Tabelle 5.32: *Anpassungsgüte des adaptiven neuronalen Netzes an die HIC-Daten unter Berücksichtigung der Capped Data-Eigenschaft beim Training.*

Tabelle 5.33 zeigt die Anpassung ohne Berücksichtigung der Capped Data-Eigenschaft beim Lernen. Zum Vergleich wurde R^2 hierbei bezüglich des Fehlers $max\left\{0, \sqrt{CAR}\right\}$ berechnet. Die Modelle lieferten schlechtere Ergebnisse auf den Test- und Trainingsdaten. Die ohnehin schlechte Extrapolation ist zwar im Vergleich zur Capped Data-Entwicklung in einigen Fällen geringfügig besser, die Unterschiede sind aber ähnlich der Schwankungen der Modelle untereinander und somit wird diesem Unterschied keine besondere Bedeutung zugemessen.

	adaptives Modell				vollständige Topologie			
Topologie	R^2_{train}	R^2_{test}	R^2_{valid}	p_{adapt}	R^2_{train}	R^2_{test}	R^2_{valid}	p_{vollst}
19-19-1	0,4	0,24	0,17	103	0,34	0,19	0,08	400
19-20-2-1	0,34	0,17	0,1	53	0,37	0,16	0	445

Tabelle 5.33: *Anpassungsgüte des adaptiven neuronalen Netzes an die HIC-Daten ohne Berücksichtigung der Capped Data-Eigenschaft beim Training.*

Abbildung 5.9 zeigt die Anpassung der Netztopologie für die transformierte Zielgröße $CAR \rightarrow \sqrt{CAR}$ basierend auf einem 19-19-1-MLP (400 Verbindungen). Die Topologie mit geringstem Testfehler wurde nach 54 Iterationen erreicht.

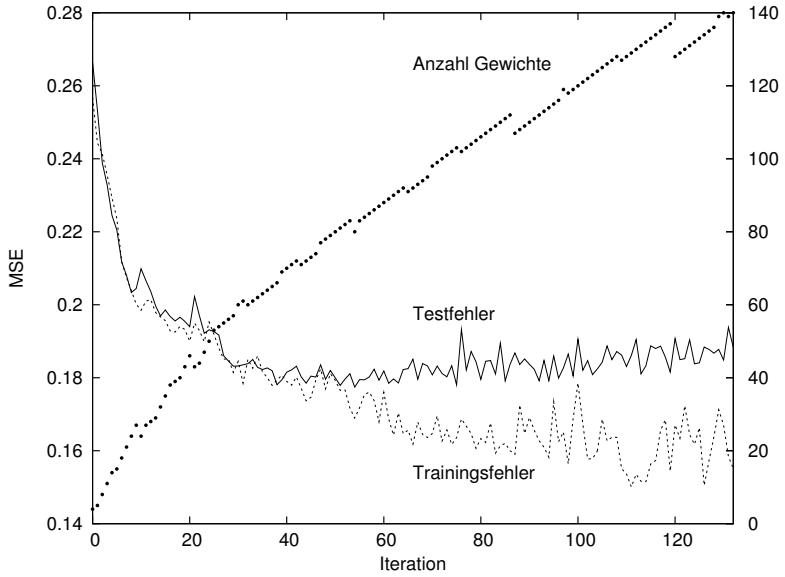


Abbildung 5.9: *Iterationsverlauf der Anpassung der Netztopologie an die HIC-Daten an eine 19-19-1-Topologie.*

5.2.6 Oszillationsmarken

Tabelle 5.34 zeigt die Anpassungsergebnisse für verschiedene Topologien von 4-4-1 (25 Gewichte) bis 4-16-8-1 (225 Gewichte). Bei kleinen Netzen brachte eine Topologieoptimierung keine Vorteile gegebüber der vollständigen Topologie. Erst bei einer hohen Anzahl Gewichte lieferte das adaptive Netz eine bessere Anpassung an Trainings- und Testdaten. In diesem Beispiel brachten Topologien mit zwei verdeckten Ebenen beim Lernen keine Vorteile gegenüber nur einer verdeckten Ebene.

	adaptives Modell			vollständige Topologie		
Topologie	R^2_{train}	R^2_{test}	p_{adapt}	R^2_{train}	R^2_{test}	p_{vollst}
4-4-1	0,51	0,53	16	0,57	0,55	25
4-8-1	0,55	0,54	35	0,55	0,49	49
4-8-4-1	0,42	0,42	53	0,57	0,45	81
4-16-1	0,62	0,65	78	0,58	0,55	97
4-16-8-1	0,58	0,53	103	0,56	0,53	225

Tabelle 5.34: *Anpassungsgüte des adaptiven neuronalen Netzes an die Oszillationsmarken-Daten.*

5.3 Bayes-Klassifikator

Im Appendix (Abschnitt 7.3) sind die Grundlagen von Bayes-Klassifikatoren erläutert. Sie sollen einen Vergleich liefern, wie genau die Daten approximiert werden können, wenn sie lediglich in Klassen eingeteilt werden. Die Einteilung in Klassen erlaubt die Zusammenfassung von nahezu gleichen Eingangsvektoren und ist daher eine realistischere Abschätzung für die bestmögliche Anpassungsfähigkeit unter Berücksichtigung des Rauschens als das scharfe Kriterium der bestmöglichen Anpassung zu identischen Eingangsvektoren (Abschnitt 1.7).

Zur Klassifizierung der Daten wurde die Zielgröße in eine vorgegebene Anzahl äquidistanter Intervalle zerlegt, von denen jedes eine Zielklasse darstellt. Um die Korrelation der Vorhersagedaten mit den Messdaten zu bestimmen, wurde jede Zielklasse durch den Mittelwert der in ihr liegenden Messwerte repräsentiert. Die zu den Zielklassen passenden Klassenintervalle der Eingangsparameter wurden durch den CAIM-Diskretisierungsalgorithmus [114] bestimmt. Dieser Algorithmus maximiert die Abhängigkeit der Zielklassen von den Eingangsklassen separat für jede Eingangsgröße bei gleichzeitiger Minimierung der Anzahl der dazu benötigten Eingangsklassen.

Zur Klassifizierung der empirischen Messdaten wurde der Diskretisierer auf die gesamte Lerndatenmenge der Eingangsgrößen angewendet, um sicherzustellen, dass keine Testdaten ausserhalb der Diskretisierungsintervalle liegen. Der Klassifikator ist nicht zur Extrapolation geeignet, da er die zu vorhersagenden Werte nur den bereits aus dem Training bekannten Klassen zuordnen kann.

Da die Benchmark-Funktionen auf die Bewertung der Generalisierungsfähigkeit bei Extrapolation abzielen, wurde der Klassifikator hierauf nicht angewendet. Die Polynommodelle und neuronalen Netze haben ohnehin auf den Benchmark-Funktionen bereits eine Modellierbarkeit von $R^2_{train} \geq 0,99$ belegt, entweder adaptiv oder durch Überanpassung vollständiger hochgradiger Polynome bzw. entsprechend langem Trainieren der Netze. Daher ist die Abschätzung einer oberen Schranke für R^2_{train} durch einen Bayes-Klassifikator für diese Funktionen nicht nötig.

5.3.1 Zugfestigkeit

Die untransformierte Zielgröße wurde in 50 äquidistante Intervalle zerlegt. Bei dem Versuch, eine noch feinere Unterteilung zu erzielen, scheiterte die Implementierung des CAIM-Diskretisierers an der Komplexität. Der WAODE-Klassifikator (Abschnitt 7.3.2) erzielte auf 50% Trainingsdaten $R^2_{train} = 0,94$ und auf 50% Testdaten $R^2_{test} = 0,85$. Durch die im Vergleich zu HIC hohe Anzahl Messdaten konnte eine feinere Unterteilung in Klassenintervalle erzielt werden, so dass die Vorhersagen für die Testdaten durch eine besser angepasste Aufteilung der Messintervalle abgedeckt wurden.

5.3.2 Wasserstoffinduzierte Rissbildung (HIC)

Die Zielgröße $\sqrt{CAR}$ wurde in 20 äquidistante Intervalle zerlegt. Der AODE-Klassifikator erzielte auf 50% Trainingsdaten $R^2_{train} = 0,81$ und auf 50% Testdaten $R^2_{test} = 0,28$. Diese Diskrepanz zwischen R^2_{train} und R^2_{test} belegt eine schlechte Generalisierungsfähigkeit.

5.3.3 Oszillationsmarken

Die $d_{OSM} \rightarrow \sqrt{d_{OSM}}$ transformierte Zielgröße wurde in zehn äquidistante Intervalle zerlegt. Das Training des AODE-Klassifikators auf 50% der Messdaten erzielte $R^2_{train} = 0,71$ und die Vorhersage der 50% Testdaten erzielte $R^2_{test} = 0,39$. Bei der Zerlegung der Zielgröße in 20 äquidistante Intervalle ergab sich für das Training $R^2_{train} = 0,67$ und für die Testdaten $R^2_{test} = 0,44$.

Kapitel 6

Zusammenfassung

Diese Arbeit befasst sich mit dem Problem, aus hochdimensionalen Datensätzen Zusammenhänge zwischen unabhängigen Eingangsgrößen und einer Zielgröße zu lernen. Bei wachsender Anzahl der Eingangsgrößen steigt die zur gleichmäßigen Abdeckung des Datenraums benötigte Anzahl der Messungen exponentiell, so dass selbst eine scheinbar hohe Anzahl Messungen zu einem unvollständig und dünn beprobten Datenraum führen kann. Auf diesen Datenraum sollen statistische Methoden angewendet werden, um ohne weiteres Vorwissen ein Maximum an Informationen herauszukristallisieren. Dabei kommt erschwerend hinzu, dass der Informationsgehalt der Messdaten durch Messfehler und Streuungen verringert oder durch Schwellwerte beschnitten wird.

Solche Eigenschaften erschweren die Modellierung erheblich und Standard-Verfahren erreichen schnell ihre Grenzen. Zwecks einer effektiven Auswertung wurden neue Verfahren zur Anpassung von Polynommodellen und neuronalen Netzen vorgestellt, die robuste Modelle generieren, mit denen globale Zusammenhänge aus den Datensätzen extrahiert werden. Diese sollen in der Lage sein, Werte in durch Messungen abgesicherten Bereichen korrekt wiederzugeben und das Gelernte auch in Gebiete fortzusetzen, die von Messdaten noch nicht abgedeckt sind.

Einfache Modelle sind meist nicht in der Lage, komplexe funktionale Zusammenhänge adäquat abzubilden, während komplexe Modelle überanpassen oder gar nicht erst erstellt werden können, insbesondere wenn die Anzahl der Modellparameter ähnlich hoch oder sogar höher ist als die Anzahl der Messungen.

Es wurde der Ansatz verfolgt, ein simples Ausgangsmodell schrittweise zu erweitern und zu vereinfachen, bis ein Modell gefunden wurde, das eine angemessene Komplexität besitzt, um die Lerndaten gut wiedergegeben werden ohne überanzupassen.

Bei den Polynommodellen wurden dazu Kandidatenmodelle aus der Summe einer Auswahl an Termen verschiedener Potenzordnungen der Eingangsgrößen erstellt. Die Bewertung der Einzelmodelle erfolgte durch ihre Generalisierungsfähigkeit auf eine beim Anpassen der Koeffizienten an die Trainings-

daten unbekannte Testdatenmenge.

Bei den neuronalen Netzen wurden Kandidatenmodelle erstellt, bei denen Verbindungen zwischen den Neuronen benachbarter Ebenen aktiviert oder deaktiviert wurden. Die Netze wurden danach mit einem Teil der Lerndaten trainiert und mittels einer disjunkten Testdatenmenge bezüglich ihrer Generalisierungsfähigkeit bewertet.

Die adaptierten Modelle wurden soweit möglich mit den entsprechenden vollständigen Modellen verglichen. Zusätzlich wurden die zu lernenden Daten mittels eines Bayes-Klassifikators ausgewertet. Dieser teilt die Daten in angepasste Klassen ein und soll ein Maß für die maximal mögliche Anpassungsfähigkeit der Trainingsdaten liefern, die aufgrund zufälliger Streuungen der Zielgröße nach oben beschränkt sein kann.

Die Beschneidung von Messwerten durch einen Schwellwert stellt ein zusätzliches Problem bei der Modellierung dar. Diese spezielle Eigenschaft wurde durch einen Algorithmus für Polynommodelle berücksichtigt, der eine Substitution der Schwellwert-Daten durch extrapolierte Vorhersagen erlaubt. Dadurch wird eine bessere Anpassung an die nicht schwellwertbehafteten Daten ermöglicht, ohne die in den Schwellwertdaten enthaltene Information zu ignorieren. Bei neuronalen Netzen wurde ein ähnlicher Ansatz durch eine modifizierte Fehlerbetrachtung ermöglicht.

Die Modelle wurden auf reale Messdatensätze verschiedener physikalischer Probleme sowie künstliche Benchmark-Datensätze angewendet. Der Vergleich der Modellergebnisse untereinander soll aber nicht verallgemeinert werden, da die Anpassungsgüte der verschiedenen Modelltypen auf unterschiedlich gearteten Datensätzen durchaus variieren kann. Jedes Modell eignet sich für bestimmte Fälle mehr oder weniger gut und ein pauschal bestes Modell existiert nicht. Die Wahl der Modellierparameter stellt einen Kompromiss aus Approximationsgüte und Zeitaufwand dar und kann ebenfalls einen großen Einfluss auf das Ergebnis haben.

Tabelle 6.1 fasst die Anpassungsgüte der Modelle für die jeweiligen Datensätze zusammen. Dargestellt ist das Bestimmtheitsmaß R^2_{train} der Abbildung der Trainingsdaten und R^2_{valid} zur Bewertung der Generalisierungsfähigkeit. Bei den Benchmark-Funktionen wurde dazu der Extrapolations-Validierungsdatensatz gewählt, bei den empirischen Funktionen ein zum Lernen nicht verwendeter Validierungsdatensatz. Es wurde aus allen Anwendungsfällen jeweils das Modell ausgewählt, das die beste Generalisierungsfähigkeit erzielte.

	Polynom, adaptiv		NN, adaptiv		WAODE	
Datensatz	R^2_{train}	R^2_{valid}	R^2_{train}	R^2_{valid}	R^2_{train}	R^2_{valid}
Benchmark 1	0,89	0,99	0,93	0,48	/	/
Benchmark 2	1	0,93	0,85	0,01	/	/
Benchmark 3	1	0,81	0,87	0	/	/
Benchmark 4	0,99	0,77	0,82	0,06	/	/
Benchmark 5	1	0,7	0,8	0,01	/	/
Benchmark 6	0,99	0,54	0,88	0,12	/	/
Benchmark 7	0,98	0,96	0,89	0,73	/	/
HIC	0,65	0,63	0,4	0,17	0,81	0,28
Zugfestigkeit	0,93	0,86	0,93	0,8	0,94	0,85
Oszillationsmarken	0,56	0,54	0,62	0,65	0,67	0,44

Tabelle 6.1: *Zusammenfassung der Anpassungsgüte der Modelle bezüglich der Benchmark- und empirischen Datensätze. R^2_{train} beschreibt die Wiedergabe der Trainingsdaten und R^2_{valid} die Generalisierungsfähigkeit bezüglich eines Validierungsdatensatzes.*

In allen Beispielen haben sowohl die adaptiven Polynommodelle als auch die adaptiven neuronalen Netze bei der Wiedergabe der Lerndaten ähnlich gut oder besser abgeschnitten als die entsprechenden vollständigen Modelle und waren bei der Extrapolation auf die Validierungsdaten durchgehend besser.

Bei der niedrigdimensionalen Benchmark-Funktion erzielten beide Arten der adaptiven Modelle sehr gute Ergebnisse auf den Lerndaten. Die Extrapolation auf außerhalb des Lernintervalls liegende Validierungsdaten war beim Polynommodell besser als bei den neuronalen Netzen. Bei höherdimensionalen Benchmark-Funktionen, insbesondere bei verrauschten Daten, waren die Polynommodelle bei Inter- und Extrapolation überlegen. Die neuronalen Netze waren praktisch nicht mehr extrapolationsfähig.

Der Zugfestigkeits-Datensatz umfasst verhältnismäßig viele Messungen, die nur schwach verrauscht sind. Sowohl die adaptiven, als auch die vollständigen Modelle konnten vergleichbar gute Ergebnisse auf den Lerndaten erzielen und generalisieren.

Die HIC-Daten sind durch starke Streuung, Informationsverlust durch Schwellwerte und wenig Messungen im Vergleich zur Anzahl der Eingangsparameter charakterisiert. Sie konnten von adaptiven Polynommodellen und neuronalen Netzen nur mäßig approximiert werden. Der Bayes-Klassifikator sowie vollständige überanpassende Polynommodelle und neuronale Netze konnten hierbei die Trainingsdaten durch Überanpassung besser wiedergeben. Die Vorhersage der Testdaten durch die Polynommodelle war besser als durch die neuronalen Netze.

Der Oszillationsmarken-Datensatz besteht aus wenigen Messungen mit starken Streuungen und wenigen Eingangsparametern. Polynommodelle und neuronale Netze konnten die Daten mit ähnlicher Approximationsgüte wiedergeben und generalisieren. Der Bayes-Klassifikator zeigte eine geringfügig bessere Anpassungsfähigkeit an die Lerndaten. Da diese drei grundlegend unterschiedlichen Modelle den Da-

tensatz jeweils ähnlich schlecht abbilden, liegt die Vermutung nahe, dass dafür drei Gründe in Frage kommen, die unabhängig vom Modell keine bessere Anpassung zulassen:

- eine hohe zufällige Streuung der Messdaten
- ein unvollständiger Satz Prozessparameter
- eine hohe Komplexität der Zielfunktion, die durch die geringe Anzahl Messdaten nicht hinreichend abgedeckt wird
- ein intrinsisch komplexer Prozess, bei dem geringfügige Änderungen in den Anfangs- oder Rahmenbedingungen zu starken Änderungen des Ergebnisses führen

Der Rechenaufwand der Adaption kann insbesondere im Fall der neuronalen Netze sehr hoch werden, wenn das Training der einzelnen Netze durch eine hohe Anzahl von Epochen oder viele parallele Instanzen gute Lösungen finden soll. Das gleiche gilt für eine hohe Anzahl an Auswahltermen, Kreuzvalidierungen und ggf. Capped Data-Schritten bei den Polynommodellen. Da in jedem Adaptionsschritt mehrere Modelle unabhängig voneinander bewertet werden müssen, bietet sich eine Parallelisierung der Algorithmen an, so dass kürzere Rechenzeiten realisiert werden können.

Die Generalisierungsfähigkeit der Modelle verbessert sich durch die Adaption der Termstrukturen bzw. Topologieen. Dem hohen Rechenaufwand der Adaptionsverfahren steht ein schnelleres und effizienteres Trainieren der kompakten adaptierten Strukturen im Vergleich zu vollständigen Modellen gegenüber. Dadurch bieten sich Vorteile beim Neutrainieren der Modelle, wenn beispielsweise neue Daten hinzugefügt werden.

Abschließend sei darauf hingewiesen, dass der Einsatz statistischer Modelle zur Approxmiation funktionaler Zusammenhänge nicht als automatische „Black box"-Lösung aufgefasst werden sollte. Die auf rein statistischen Annahmen ermittelten Zusammenhänge zwischen Eingangsparametern und Zielgröße sind nicht notwendigerweise kausaler Natur, wenn die Daten nicht mit Bedacht ausgewählt wurden. So kann z. B. eine scheinbare Korrelation zwischen einer Eingangsgröße x_i und der Zielgröße y gefunden werden, obwohl tatsächlich gar kein Zusammenhang besteht. Dies kann passieren, wenn y nicht von x_i, sondern mit einem anderen x_j abhängt, das mit x_i korreliert ist, selbst aber nicht im Datensatz berücksichtigt wurde. Es ist daher für eine effektive Anwendung der Modelle notwendig, dass sowohl bei der Auswahl und Zusammenstellung der Daten, als auch bei der Interpretation der Ergebnisse so viel metallurgisches und physikalisches Expertenwissen wie möglich einfließt.

Kapitel 7

Appendix

7.1 Multiple Polynomregression

7.1.1 Grundlagen

Bei der multiplen Polynomregression werden die Eingangsdaten durch ein Polynom $f(x_1,\ldots,x_m)$ approximiert, so dass die Fehlersumme

$$\sum_{i=1}^{n}\left\|y_i - f\Big(\big(x_1,\ldots,x_m\big)_i\Big)\right\| \tag{7.1.1}$$

bezüglich einer Norm $\|\cdot\|$ minimiert wird. Eine gängige Methode ist die der kleinsten Fehlerquadrate [24], bei der die euklidische Norm $\|\cdot\|_2$ verwendet wird. Im Folgenden wird eine Regression verwendet, die linear in ihren Koeffizienten ist und nichtlineare Abhängigkeiten der Eingangsparameter $x_1,\ldots,x_m$ erlaubt:

$$f(x_1,\ldots,x_m) = \sum_i a_i z_i(x_1,\ldots,x_m) \tag{7.1.2}$$

Hierbei bezeichnen die a_i die zu bestimmenden Koeffizienten und die Regressionsvariablen $z_i(x_1,\ldots,x_m)$ sind Funktionen der Eingangsparameter, die keine weiteren Unbekannten enthalten. Auf diese Weise können vorab bekannte Abhängigkeiten der Zielgröße mit linearen Parametern durch entsprechende Wahl der z_i explizit eingebaut werden. Im Beispiel $y = a\frac{1}{x}$ wäre dies direkt durch Bestimmung der Koeffizienten a zum Regressor $z \equiv \frac{1}{x}$ oder durch geeignete Transformation eines multiplikativen Systems in ein additives wie im Fall $y = a\exp(bx) \Leftrightarrow \ln y = \ln a + bx$ möglich. Bei der Transformation der Variablen ist allerdings zu beachten, dass die Fehler mittransformiert werden. Dies kann zu einer verzerrten Gewichtung

der Fehlernorm führen. Falls die Fehlerverteilung durch die Transformation in besserer Näherung symmetrisch verteilt wird als im untransformierten Fall, ist diese Transformation sinnvoll. Andernfalls wird das System besser direkt mithilfe eines nichtlinearen Approximationsverfahrens, z.B. des Levenberg-Marquardt-Verfahrens [24], gelöst.

Das additive Gleichungssystem wird im Folgenden in Matrixschreibweise behandelt:

$$Aa = y \tag{7.1.3}$$

$$A = \begin{pmatrix} 1 & z_1(x_1,\ldots,x_m)_1 & z_2(x_1,\ldots,x_m)_1 & \cdots & z_p(x_1,\ldots,x_m)_1 \\ 1 & z_1(x_1,\ldots,x_m)_2 & z_2(x_1,\ldots,x_m)_2 & \cdots & z_p(x_1,\ldots,x_m)_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & z_1(x_1,\ldots,x_m)_n & z_2(x_1,\ldots,x_m)_n & \cdots & z_p(x_1,\ldots,x_m)_n \end{pmatrix}$$

$$a = \begin{pmatrix} a_{konst} \\ a_1 \\ a_2 \\ \vdots \\ a_p \end{pmatrix} \qquad y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}.$$

A ist die Matrix der Regressionsvariablen, $z_i(x_1,\ldots,x_m)_j$ sind Funktionen der Eingangsparameter, a der Vektor der Regressionsparameter und y der Lösungsvektor. Die Spalte $(1,\ldots,1)^T$ von A berücksichtigt dabei die additive Konstante *konst* der Regression.

Die Bedingung $\sum_{i=1}^{n} \|y - Aa\| \stackrel{!}{=} \min$ kann durch partielles Ableiten bezüglich der a_i und Nullsetzen in die Form $A^T Aa = A^T y$ überführt werden. Dies ist die Ordinary Least Squares-Methode. Die symmetrische positiv definite Matrix $A^T A$ erlaubt eine Lösung des Gleichungssystems beispielsweise per Cholesky-Zerlegung [24] oder dem Verfahren der konjugierten Gradienten. Diese Methoden sind aber nicht stabil bezüglich schlecht konditionierter Matrizen, so dass sie für die in vielen Fällen quadratisch schlecht konditionierte Matrix $A^T A$ nicht geeignet sind. Zur Umsetzung der folgenden Algorithmen wurde eine *QR*-Zerlegung der Matrix A verwendet [115]. Hierbei wird die Matrix A in ein Produkt aus einer orthogonalen Matrix Q ($Q^T Q = I$) und einer oberen Dreiecksmatrix R zerlegt. Damit lässt sich das Glei-

chungssystem durch Rückwärtssubstitution und Matrixmultiplikation lösen:

$$\begin{aligned} Aa &= y \\ QRa &= y \\ Ra &= Q^T y, \, a \text{ folgt aus Rückwärtssubstitution.} \end{aligned} \tag{7.1.4}$$

Der in dieser Arbeit vorgestellte Algorithmus wurde in C++ umgsetzt. Zur Lösung der Matrixgleichungen wurde die newmat11-Bibliothek [116] verwendet.

Eine Regression kann schrittweise durchgeführt werden, indem man zunächst mit einer linearen Regression beginnt. Erhält man hier schon eine gute Anpassung, lassen sich die Ergebnisse leicht interpretieren. Die Beträge der Steigungen der Regressionsgeraden der normierten Eingangsparameter sind dann im Rahmen des Modells ein Maß für die Korrelation der Parameter mit der Zielgröße. In den Residuen Messdaten - Modelldaten lassen sich Nichtlinearitäten erkennen, die dann im nächsten Regressionsansatz durch eine geeignete Transformation der Eingangsparameter oder die Ergänzung des Modells um höhergradige Terme berücksichtigt werden können.

Oft läßt sich die Zielgröße aber nicht lediglich durch eine Summe unabhängiger Einzelbeiträge der Eingangsparameter beschreiben, sondern es treten Abhängigkeiten von Kombinationen aus ihnen auf. So lässt sich beispielsweise der Treibstoffverbrauch eines Autos durch ein Modell, das Geschwindigkeit v und Entfernung d lediglich additiv berücksichtigt, offensichtlich nicht sinnvoll beschreiben. Wenn der Treibstoffverbrauch näherungsweise quadratisch mit v und linear mit d wächst, ist die Wechselwirkungsgröße $v^2 d$ besser geeignet. Zur Beschreibung von Wechselwirkungen können Kreuzterme der Form $x_i \cdot x_j$ herangezogen werden.

Bei einer linearen Regression mit Kreuztermen hängt die Steigung der Funktion nach einem Eingangsparameter x_q von den restlichen Eingangsparametern ab

$$\begin{aligned} \frac{\partial f(x_1,\ldots,x_m)}{\partial x_q} &= \frac{\partial}{\partial x_q}\left(konst + \sum_{i=1}^{m} a_i x_i + \sum_{\substack{i,j=1 \\ j>i}}^{m} b_{ij} x_i x_j \right) \\ &= a_q + \sum_{j=1}^{q-1} b_{jq} x_j + \sum_{j=q+1}^{m} b_{qj} x_j \\ &= a_q + \phi\left(x_1,\ldots,x_{q-1},x_{q+1},\ldots,x_m\right) \end{aligned} \tag{7.1.5}$$

so dass in diesem Fall allgemein keine globale Aussage über dessen Einfluss auf die Zielgröße gemacht werden kann, sondern lediglich um diskrete Punkte $(x_1,\ldots,x_{q-1},\ x_{q+1},\ldots,x_m)$. Diese Abhängigkeit gilt

entsprechend für erweiterte Kreuzterme der Form $\prod_i \phi_i(x_i)$, wobei die ϕ_i die Beschreibung von Nichtlinearitäten ermöglichen. Analog dazu hängt die Form einer nichtlinearen Anpassungskurve bezüglich eines Eingangsparameters von der Wahl der restlichen Eingangsparameter ab.

7.1.2 Berechnungsalgorithmus

Der zur Auswertung der Messdaten umgesetzte Algorithmus erlaubt die Anpassung durch Least Squares-Regressionen verschiedener polymomieller Ordnungen mit Wechselwirkungen der Eingangsparameter. Ein solcher Ansatz ist gerechtfertigt, falls die Zielgröße eine hinreichend glatte Funktion der Eingangsparameter ist. Unstetigkeiten könnten lediglich durch hochgradige Polynome approximiert werden. Aufgrund ihrer Tendenz zur Oszillation bei verrauschten Daten würde dies aber zu unbrauchbaren Ergebnissen führen. In solchen Fällen sollten die Daten in geeignete Klassen aufgeteilt und diese Klassen einzeln modelliert werden.

In Anlehnung an eine Taylor-Entwicklung wird die Zielfunktion durch Polynome geeigneten Grades approximiert. Folgende Aufteilung stellt das Regressionspolynom in zweckmäßiger Form dar:

$$
\begin{aligned}
f(x_1,\ldots,x_m) \quad = konst \quad & + \overbrace{\sum_{i=1}^{m} a_i x_i + \sum_{i=1}^{m} b_i x_i^2 + \sum_{i=1}^{m} c_i x_i^3 + \ldots}^{\text{Einzelwirkungen}} \\
& + \overbrace{\sum_{\substack{i,j=1 \\ j>i}}^{m} d_{ij} x_i x_j + \sum_{\substack{i,j,k=1 \\ k>j>i}}^{m} e_{ijk} x_i x_j x_k + \ldots}^{\text{lineare Kreuzterme}} \\
& + \overbrace{\sum_{\substack{i,j=1 \\ j>i}}^{m} f_{ij} x_i^{\alpha} x_j^{\beta} + \sum_{\substack{i,j,k=1 \\ k>j>i}}^{m} g_{ijk} x_i^{\gamma} x_j^{\delta} x_k^{\varepsilon} + \ldots}^{\text{höhergradige Kreuzterme}}, \\
& \{\alpha,\beta,\gamma,\delta,\varepsilon\} \in \mathbb{N} \geq 2
\end{aligned}
\tag{7.1.6}
$$

Anpassungen der Ordnung 3 spielen in dieser Arbeit eine besondere Rolle, da sie als Basis für adaptive Polynome bei vielen der behandelten Probleme gute Ergebnisse lieferten. Sie lassen sich folgendermaßen interpretieren:

$$
\begin{aligned}
f(x_1,\ldots,x_m) \quad = \quad & \\
& \left\{
\begin{array}{ll}
\sum_{i=1}^{m} a_i x_i & \text{(a)} \\
\text{(1. Ordnung)} & \\
 & \\
\sum_{i=1}^{m} a_i x_i + \sum_{i=1}^{m} b_i x_i^2 + \sum_{\substack{i,j=1 \\ j>i}}^{m} d_{ij} x_i x_j & \text{(b)} \\
\text{(2. Ordnung, paarweise Kreuzterme)} & \\
 & \\
\sum_{i=1}^{m} a_i x_i + \sum_{i=1}^{m} b_i x_i^2 + \sum_{i=1}^{m} b_i x_i^3 + & \text{(c)} \\
\sum_{\substack{i,j=1 \\ j>i}}^{m} d_{ij} x_i x_j + \sum_{\substack{i,j,k=1 \\ k>j>i}}^{m} e_{ijk} x_i x_j x_k + & \\
\sum_{\substack{i,j=1 \\ j>i}}^{m} f_{ij} x_i^2 x_j & \\
\text{(3. Ordnung, paar- und tripelweise Kreuzterme)} &
\end{array}
\right. \\
konst \quad + & \qquad (7.1.7)
\end{aligned}
$$

Gleichung 7.1.7 (a) eignet sich, um einfache lineare Trends abzuschätzen. Die quadratischen Terme in Gleichung 7.1.7 (b) erlauben eine Approximation von Krümmungen der einzelnen x_i in einer globalen Abhängigkeit, d.h. invariant bezüglich der restlichen $x_{j\neq i}$. Die paarweisen Kreuzterme 1. Ordnung erlauben eine Approximation linearer Wechselwirkungen zweier Eingangsparameter. Die kubischen Terme in 7.1.7 (c) erlauben eine Modellierung von Krümmungen und von Wendepunkten. Die tripelweisen Kreuzterme approximieren Wechselwirkungen zwischen jeweils drei Eingangsparametern. Die Terme $x_i^2 x_j$ erlauben eine erste Approximation nichtlinearer Wechselwirkungen. Dieses Modell dritten Grades kann allerdings aufgrund der hohen Anzahl der aus den m Eingangsparametern kombinierten p freien Modellparameter schnell zu einem niedrigem Verhältnis $\frac{n}{p}$ zu den n Datenpunkten führen und läuft daher Gefahr, gerade bei stark verrauschten Messdaten überanzupassen: bei $m = 3$ Eingangsparametern erhält man $p = 19$ freie Modellparameter, bei $m = 10$ bereits $p = 285$, bei $m = 20$ $p = 1770$ und bei $m = 30$ $p = 5456$.

7.2 Künstliche neuronale Netze

7.2.1 Funktionsweise

Ein künstliches neuronales Netz ist eine Menge von untereinander verbundenen speziellen Funktionen, den Neuronen. Dabei nimmt jedes Neuron die Ausgaben der mit seiner Eingangsseite verbundenen Neuronen auf, wertet diese aus und leitet sie an die mit seiner Ausgangsseite verbundenen Neuronen weiter. Die Eingangsgrößen bilden dabei die erste Ebene des Netzes und das bzw. die letzten Neuronen bilden die Ausgabe [28, 117].

Die Verbindungen w_{ij} zwischen den Neuronen i und j sind gewichtet. Die Gewichte sind die freien Modellparameter und stellen somit das Wissen des Netzes dar. Die Auswertung eines Neurons j wird durch die Propagierungs- und die Aktivierungsfunktion realisiert. Die Propagierungsfunktion gewichtet die Ausgaben Aus_i aller Eingangsneuronen i zum Neuron j und die Aktivierungsfunktion erzeugt daraus den Ausgabewert Aus_j des Neurons. Üblicherweise wird für die Propagierungsfunktion die gewichtete Summe aus Ausgabe der Eingabeneuronen und den Verbindungen der Gewichte

$$Ein_j = \sum_i Aus_i \cdot w_{ij} \tag{7.2.1}$$

verwendet. Die Aktivierungsfunktion f_{act} berechnet daraus den Ausgabewert des Neurons

$$Aus_j = f_{act}(Ein_j) \tag{7.2.2}$$

und soll den Schaltvorgang eines biologischen Neurons simulieren, das eine Ausgabe (einen Reiz) produziert, sobald ein gewisser Schwellwert überschritten wird. Die Wahl der nichtlinearen Aktivierungsfunktion ist prinzipiell beliebig. Generell können für alle Neuronen auch unterschiedliche Propagierungs- und Aktivierungsfunktionen gewählt werden. Die Wahl dieser Funktionen sowie ihrer Aktivierungsschwellen und ggf. weiterer Funktionsparameter würde dann ebenfalls einen Teil des Wissens des Netzes repräsentieren. In vielen praktischen Fällen ist es aber ausreichend, eine globale sigmoidale Aktivierungsfunktion zu definieren.

Beispiele sind

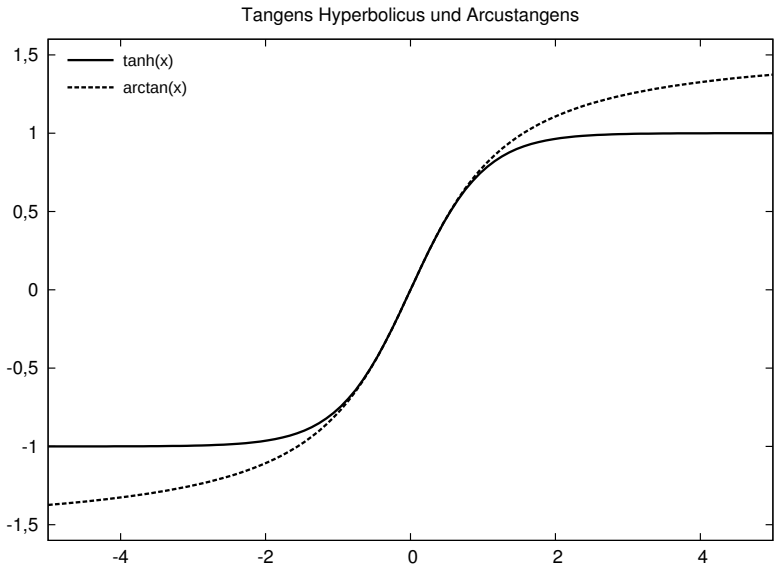
- die Fermi-Funktion $\frac{1}{1+e^{-\frac{x}{T}}}$ mit dem „Temperatur"parameter T, der die Breite des Aktivierungsbereichs definiert und dem Wertebereich $(0, 1)$
- der Tangens Hyperbolicus $\tanh(x)$ mit dem Wertebereich $(-1, 1)$ und

- der Arcustangens atan(x) mit dem Wertebereich $(-\frac{\pi}{2}, \frac{\pi}{2})$,

siehe Abbildung 7.1

Diese Funktionen haben den Definitionsbereich $(-\infty, \infty)$ und sind differenzierbar, was Voraussetzung für einige Lernalgorithmen ist.

a)



b)

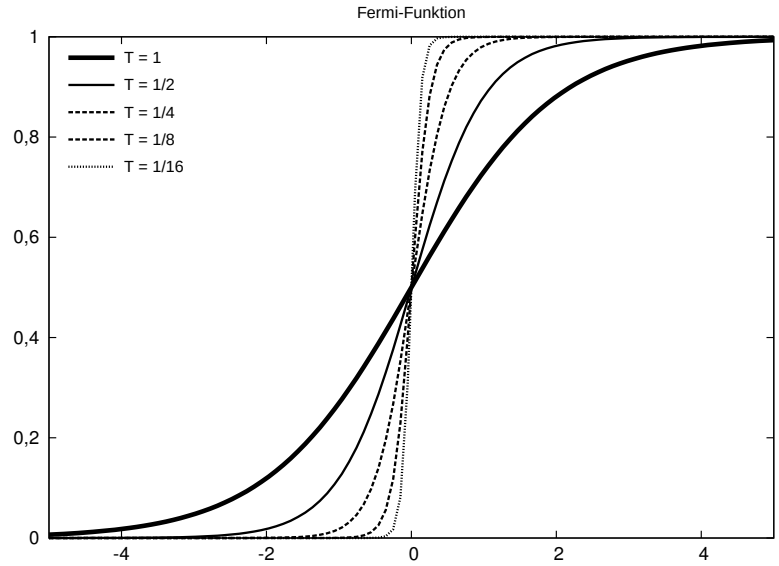


Abbildung 7.1: *Aktivierungsfunktionen a)* tanh(x) *und* atan(x) *und b) Fermi-Funktion für verschiedene Temperaturparameter.*

Die Aktivierungsfunktion der Eingabeneuronen ist die Identität. Für Funktionsapproximationen hat sich in vielen Fällen der Tangens Hyperbolicus als zweckmäßige Aktivierungsfunktion der folgenden

Neuronen erwiesen, da er sowohl positive, als auch negative Werte zurückliefern kann. Für die Ausgabeneuronen sollte eine lineare Aktivierungsfunktion gewählt werden, falls ein unbegrenztes Ausgabeintervall erzeugt werden soll.

Ein Schwellwert kann optional durch Hinzufügen von sogenannten „Bias-Neuronen" in die Eingabe realisiert werden. Bias-Neuronen erhalten keine Eingabe und geben immer den selben Schwellwert aus.

7.2.2 Netztopologien

Die Netztopologie definiert, aus wie vielen Neuronen das Netz besteht und wie sie untereinander verbunden sind.

Ein Multi Layer Perceptron (MLP) ist ein Feedforward-Netz (Abbildung 7.2), bei dem die Neuronen in nacheinander angeordneten Ebenen liegen und die Verbindungen unidirektional von Neuronen einer Ebene zu Neuronen der Folgeebene laufen. Entsprechend liegen die Verbindungen zwischen zwei Neuronenebenen in einer eigenen Verbindungsschicht. Da die Verbindungen durch ihre Gewichte dargestellt werden, werden diese auch als Gewichtsschichten bezeichnet. Man unterscheidet dabei

- die Eingabeebene, deren Neuronen die Messdaten $x_1, \ldots, x_m$ als Eingabe erhalten,
- die verdeckten Ebenen, in denen die Informationen weiter verarbeitet werden und
- die Ausgabeebene, deren Neuron(en) das Modellergebnis $f_w(x_1, \ldots, x_m)$ bezüglich der Gewichte w repräsentieren.

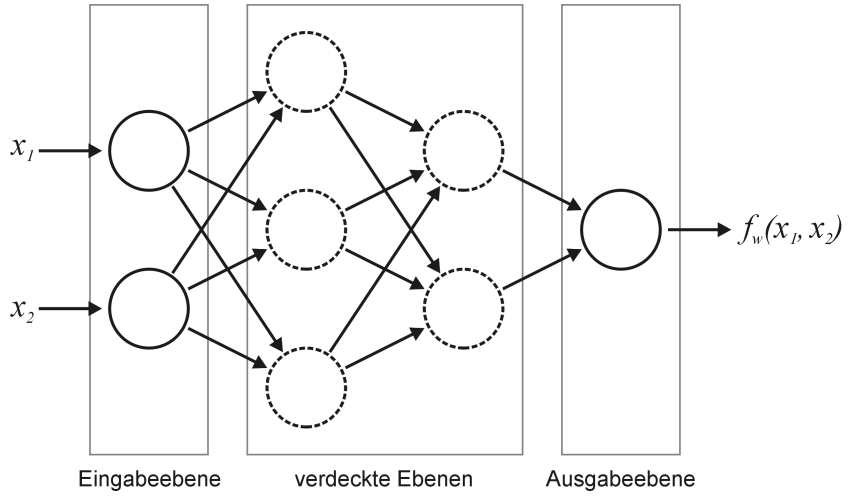


Abbildung 7.2: *Multi Layer Perceptron mit zwei verdeckten Ebenen. Die Kreise stellen die Neuronen dar und die Pfeile die gewichteten Verbindungen w.*

Das Feedforward-Netz kann erweitert werden, indem Verbindungen zugelassen werden, die eine oder mehrere Ebenen überspringen. Diese Art der Verknüpfung sowie Rückkopplungstopologien, bei denen ein Netz die Ausgabe in mehreren Schritten berechnet und dabei die Ausgaben eines Neurons im Folgeschritt als zusätzliche Eingabe verwendet, werden im Rahmen dieser Arbeit nicht betrachtet. Solche Rückkopplungen eignen sich beispielsweise für die Approximation von Zeitreihen.

Die Fähigkeit eines neuronalen Netzes, komplexe Funktionen approximieren zu können, steigt mit der Anzahl seiner Neuronen und der Verbindungen. Mit der Komplexität des Netzes steigt ebenfalls die Gefahr der Überanpassung. Die Wahl einer angepassten Topologie hängt vom Problem und der Charakteristik der zu approximierenden Daten ab. Grundsätzlich orientiert sich die Anzahl der Neuronen an der Anzahl der freien Parameter des Problems. Da diese aber nur selten bekannt ist, ist es dem Anwender überlassen, die Netztopologie zu wählen. Allgemeingültige Regeln für die optimale Netzwahl gibt es nicht. [118] empfiehlt für Perceptrons mit zwei verborgenen Ebenen mindestens $p_1 = m+1$ Neuronen in der ersten und höchstens $p_2 = p_1/3$ Neuronen in der zweiten verdeckten Ebene.

Um die Zielfunktion möglichst gut repräsentieren zu können und gleichzeitig die Gefahr der Überanpassung gering zu halten, sollte zunächst ein Netz mit wenig Neuronen evaluiert und so lange sukzessive durch komplexere Netze ersetzt werden, bis sich die Verallgemeinerungsfähigkeit nicht mehr verbessert.

Ein Perceptron ohne verdeckte Ebenen kann nur linear separierbare Probleme abbilden [28]. Durch Hinzufügen einer verdeckten Neuronenebene kann es bereits komplexe Polygone beschreiben und somit Funktionen mit endlich vielen Unstetigkeitsstellen inklusive der ersten Ableitung beliebig genau approximieren [119]. Das Hinzufügen einer weiteren Neuronenebene ermöglicht Vereinigungen und Schnitte komplexer Polygone. Dadurch kann jede beliebig verteilte Menge klassifiziert werden.

7.2.3 Lernverfahren

Für eine festgelegte Netztopologie mit vorgegebener Aktivierungsfunktion besteht das Lernen aus dem Optimieren der Verbindungsgewichte. Im Folgenden wird das Prinzip des überwachten Lernens verfolgt.

Dabei berechnet das Netz das Modellergebnis $f_w(x_1,\ldots,x_m)_i$ zum i-ten gemessenen Eingabevektor $(x_1,\ldots,x_m)_i$ und vergleicht es mit dem dazugehörigen Messwert y_i der Zielgröße. Im Lernschritt wird eine Korrektur der Gewichte berechnet, so dass sich der Fehler $Err\left(y_i, f_w\left(x_1,\ldots,x_m\right)_i\right)$ verringert. Dem Netz werden nacheinander alle Eingabevektoren der Trainingsmenge präsentiert und der dazugehörige Fehler berechnet. Der Lernschritt kann entweder unmittelbar nach jeder Eingabe mithilfe des Einzelfehlers erfolgen oder erst nach Durchlaufen der gesamten Trainingsmenge bezüglich des kumulierten Gesamtfehlers (Batch-Lernen). Das Korrigieren nach jedem Eingabevektor bietet sich an, wenn das Wissen des Netzes ständig um neue Lernwerte erweitert wird. Aufgrund der größeren Anzahl an Korrekturschritten und der Abhängigkeit des Lernfortschritts von der Präsentationsreihenfolge der Eingabevek-

toren eignet sich dieses Verfahren aber weniger für eine feste Menge an Trainingsdaten, weshalb für die hier behandelten Probleme das Batch-Lernen verwendet wurde. Eine Fehlerpropagation des Netzes für alle Eingangsvektoren wird Lernepoche genannt. Als kumuliertes Fehlermaß wird die Fehlersumme bezüglich einer Norm, wie in Gleichung 7.1.1 definiert, verwendet.

Im Gegensatz zu den linearen Regressionsalgorithmen existiert für neuronale Netze kein direktes Berechnungsverfahren zur Bestimmung der global optimalen Gewichte. Statt dessen existiert eine Vielzahl von iterativen Lernverfahren, bei denen die Gewichte in jedem Lernschritt iterativ verbessert werden.

7.2.4 Lernbarkeit

Ein Perceptron mit einer Schicht verdeckter Neuronen kann jede Funktion beliebig genau approximieren, wenn die verdeckte Schicht nur genügend Neuronen enthält. Neben der Repräsentierbarkeit ist für praktische Anwendungen aber auch die Lernbarkeit bestimmend und diese hängt von der Komplexität der Fehlerfunktion ab. Die Fehlerfunktion wird umso zerklüfteter und damit schwieriger zu optimieren, je mehr Neuronen und je mehr Schichten das Netz enthält. Bei vielen Problemen würde zwar eine verdeckte Schicht bereits zur Repräsentation des Problems ausreichen, das Verteilen der Neuronen auf zwei Schichten kann aber zu einer einfacheren Fehlerfunktion führen und schneller lernbar sein. Mehr als drei Schichten führen in der Praxis dagegen oft zu einer Fehlerfunktion mit sehr vielen lokalen Minima.

Im Folgenden werden einige Lernverfahren beschrieben, die für die Algorithmen in dieser Arbeit verwendet wurden.

7.2.4.1 Backpropagation of error

Backpropagation of error ist ein Gradienten-Abstiegsverfahren, das den Fehler als Funktion des Vektors der Netzgewichte minimiert [120]. Dabei wird von der Ausgabeebene beginnend der Fehlerbeitrag jeder Verbindung ermittelt und ein Korrekturschritt in Richtung des lokal stärksten Abstiegs berechnet.

Für Ausgabeneuronen j ist die Schrittweite gegeben durch das Produkt aus

- Ausgabe des Vorgängerneurons Aus_i,
- Gradient der Aktivierungsfunktion an der Eingabe des Ausgabeneurons $\frac{\partial f_{act}(Ein_j)}{\partial(Ein_j)}$,
- Differenz aus Zielgröße und Ausgabe des Ausgabeneurons $y - Aus_j$ und
- einer vom Benutzer festgelegten Lernrate η.

Für verdeckte Neuronen ist die Schrittweite gegeben durch das Produkt aus

- Ausgabe des Vorgängerneurons Aus_i,

- Gradient der Aktivierungsfunktion an der Eingabe des Nachfolgeneurons $\frac{\partial f_{act}(Ein_j)}{\partial(Ein_j)}$,
- der gewichteten Summe der Gewichtsänderungen zu direkt nachfolgenden Neuronen k und
- einer vom Benutzer festgelegten Lernrate η.

Damit gilt für die Gewichtsänderung Δw_{ij} zwischen den Neuronen i und j

$$\Delta w_{ij} = \eta \cdot \delta_j \cdot Aus_i$$
$$\text{mit } \delta_j = \left\{ \begin{array}{c} \frac{\partial f_{act}(Ein_j)}{\partial(Ein_j)} \cdot \left(y - Aus_j\right), \text{ falls } j \text{ Ausgabeneuron} \\ \frac{\partial f_{act}(Ein_j)}{\partial(Ein_j)} \cdot \sum_k \delta_k w_{jk}, \text{ falls } j \text{ verdecktes Neuron} \end{array} \right\} \tag{7.2.3}$$

Anschließend wird die nächste Gewichtsschicht in Richtung Eingabeebene unter Berücksichtigung der bereits korrigierten Gewichte behandelt, bis schließlich alle Gewichte korrigiert wurden.

Kleine Werte von η führen dazu, dass sich die Gewichte nur langsam ändern und das Lernverfahren entsprechend lange dauert. Bei großem η besteht die Gefahr, dass Optima übersprungen werden. Folglich hängt die optimale Wahl der Lernrate von den Eigenschaften der Fehlerfunktion und damit vom Problem selbst ab. Oftmals liefern Werte von $0,01 \leq \eta \leq 0,9$ akzeptable Lernergebnisse.

Da sich der Fehler zu Beginn der Lernphase schneller verringert als in der Nähe eines Optimums, bietet es sich an, die Lernrate im Verlauf des Lernens zu verringern. Das sollte aber nicht kontinuierlich erfolgen, sondern stufenweise. Andernfalls kann es vorkommen, dass die Verbesserungsschritte durch den Abfall der Lernrate so stark ausgebremst werden, dass die Steigung vor einem zu überwindenden lokalen Maximum der Fehlerfunktion nicht mehr überwunden werden kann.

Da die Gewichtsschichten nahe der Ausgabeschicht als erste korrigiert werden, lernen sie am schnellsten. Daher kann es sinnvoll sein, für die Gewichtsschichten nahe der Eingabeschicht höhere Lernraten zu wählen.

Beispiel

An einem einfachen neuronalen Netz ohne versteckte Ebenen soll ein Backpropagation-Schritt demonstriert werden (Abbildung 7.3). Es soll die Summe der beiden in die Neuronen *N1* und *N2* eingegebenen Werte gebildet werden, indem die Gewichte w_{13} zwischen *N1* und dem Ausgabeneuron *N3* und w_{23} zwischen *N2* und *N3* trainiert werden.

Im Beispiel werden dem Netz dazu die Lerndaten $\{(x_1 = 1, x_2 = 2, y = 3)_1, (x_1 = 3, x_2 = 4, y = 7)_2\}$ präsentiert. Als Lernrate wird $\eta = 0,03$ gewählt und als Aktivierungsfunktion die Identität $f_{act}(x) = \mathrm{id}_x$. In diesem Trivialbeispiel ist klar, dass beide Gewichte 1 sein müssen.

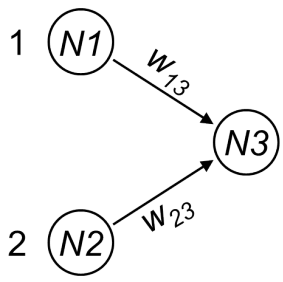


Abbildung 7.3: *Einfaches Beispielnetz*

Mit einer Initialisierung der Gewichte $w_{13} = 0$ und $w_{23} = 0$ ergeben sich die Netzausgaben $N1 \cdot w_{13} + N2 \cdot w_{23}$ zu $Aus(N3) = 0$.
Die Backpropagation-Schritte zur Aktualisierung der Gewichte lauten mit $\partial f_{act}(Ein_j)/\partial(Ein_j) = 1$ (Gleichung 7.2.3):

$$\triangle w_{i3} = \eta \cdot \sum_{k=1}^{2} \left(y_k - Aus_{(k)}(N3)\right) \cdot Aus_{(k)}(Ni) \tag{7.2.4}$$

wobei $Aus_{(k)}(Ni)$ die Ausgabe des Neurons i zum k-ten Lerntupel bezeichnet.
Damit ergibt sich im ersten Schritt $\triangle w_{13} = 0,72$ und $\triangle w_{23} = 1,02$.
Mit der Aktualisierung $w_{ij}^{neu} = w_{ij}^{alt} + \triangle w_{ij}$ folgt für die Eingabe $\{1, 2\}$ die Ausgabe $Aus(N3) = 2,76$ und für $\{3, 4\}$ $Aus(N3) = 6,24$.
Diese simple Form der Backpropagation konvergiert mit diesen Lernparametern nur langsam. Nach 500 Lernschritten ergibt sich erst $w_{13} = 0.974$ und $w_{23} = 1,019$.

7.2.4.2 Verbesserungen zu Backpropagation

Die Effektivität von Gradientenabstiegsmethoden kann stark eingeschränkt werden, wenn die Annahme nicht erfüllt ist, dass der Abstand zum Optimum umso größer ist, je größer der Fehlergradient ist. Bei den üblicherweise komplexen Fehlerfunktionen neuronaler Netze kommt dieser Fall oft vor. Um dem entgegenzuwirken, wurden mehrere Erweiterungen des Backpropagation-Algorithmus entwickelt, von denen einige hier vorgestellt werden.

Trägheitsterm

Die Wahl der Schrittweite proportional zum Betrag des Gradienten ist angemessen, wenn die Fehlerfunktion konvex ist. Dies ist nahe eines lokalen Optimums der Fall. In größerer Entfernung zu einem Optimum kann die Fehlerfunktion aber konkav sein oder sich kaum ändern. Auf einem solchen Plateau kann das Lernverfahren aufgrund des fast verschwindenden Gradienten stagnieren. Um dies zu verhindern, wird

zur Gewichtsänderung $\triangle w_\alpha$ im Lernschritt α ein Teil der Gewichtsänderung des vorhergehenden Schritts hinzuaddiert, $\triangle w_\alpha \rightarrow \triangle w_\alpha + p \cdot \triangle w_{\alpha-1}$ [120]. Dieser Term mit dem Faktor $0 < p < 1$ soll der Gewichtsänderung eine Trägheit geben, um dem schnellen Abbremsen auf einem Plateau entgegenzuwirken. Er kann weiterhin Oszillationen um ein Minimum dämpfen. Gerade im hochdimensionalen Fall bei stark zerklüfteten Fehlerflächen kann dieser Effekt aber auch ins Gegenteil umschlagen und zum Verlassen eines Minimums führen.

Flat spot elimination

Die Ableitung sigmoidaler Aktivierungsfunktionen ist in größerer Entfernung zum Aktivierungsbereich fast null. Entsprechend langsam reagiert das Lernverfahren auf Neuronen, die eine Eingabe weit außerhalb dieses Bereichs erhalten. Um dies zu verhindern, kann die Ableitung der Aktivierungsfunktion z. B. durch Addition einer Konstante modifiziert werden [121].

Weight decay

Weight decay verhindert, dass einzelne Gewichte stark anwachsen, indem der Fehlerterm um die Summe der Gewichtsquadrate $Err \rightarrow Err + \frac{1}{2}\beta \sum_{i,,j} w_{i,j}^2$ ergänzt wird [122]. Sehr große Gewichte zu verdeckten Neuronen können dazu führen, dass die Netzausgabe zu grob wird. Bei Ausgabeneuronen können sie starke Schwankungen in der Netzausgabe zur Folge haben.

Höhergradige Approximationen der Fehlerfunktion

Bei der lokalen Approximation der Fehlerfunktion durch eine Parabel und direkte Berechnung deren Minimums kann die Konvergenzgeschwindigkeit gegenüber der des Gradientenabstiegs erhöht werden. Dies wird beispielsweise beim Quickprop-Algorithmus [121] benutzt. Das Verfahren ist allerdings relativ komplex und erfordert die Berücksichtigung vieler Sonderfälle. Es konvergiert schlechter als Standard-Backpropagation, wenn die Fehlerfunktion nicht parabolisch approximiert werden kann.

7.2.4.3 Initialisierung der Gewichte

Da das Backpropagation-Verfahren lediglich in Richtung einer Verbesserung iteriert, konvergiert es je nach Wahl der Lernparameter in vielen Fällen gegen ein lokales Optimum in der Nähe des durch die Anfangsgewichte definierten Startpunkts. Insbesondere kann die Konvergenz zu einem globalen Optimum nicht garantiert werden.

Die Gewichte können nicht alle mit dem gleichen Wert initialisiert werden, da sie dann im Training immer gleichermaßen geändert würden. Dieses Problem kann durch die Initialisierung mit Zufallszahlen

gelöst werden. Das Intervall sollte aber nicht zu groß gewählt werden, da das Lernen sonst durch große Einzelgewichte verlangsamt werden kann oder frühzeitig in einem ungeeigneten Nebenminimum endet.

7.2.4.4 Resilient Backpropagation of error (Rprop)

Resilient Backpropagation ist ein Batch-Learning Verfahren, das weniger Annahmen bezüglich des Verhaltens der Fehlerfunktion als Backpropagation mit Gradientenabstieg macht. Dadurch werden einige für diese Art von Algorithmus typische Probleme umgangen.

Der Rprop-Algorithmus berücksichtigt lediglich die Richtung des steilsten Abstiegs und nicht den Betrag des Gradienten der Fehlerfunktion [123]. Die Schrittweite wird dabei jedes mal um einen vorgegebenen Faktor $\eta^+ > 1$ vergrößert, wenn sich das Vorzeichen der partiellen Ableitung der Fehlerfunktion nicht ändert. Falls sich das Vorzeichen ändert, was bedeutet, dass ein Minimum übersprungen wurde, wird die Schrittweite um einen Faktor $\eta^- < 1$ verringert. Für die maximale und minimale Schrittweite Δ_{max} und Δ_{min} wird eine Grenze vorgegeben. Dieser Algorithmus konvergiert bei vielen Problemen deutlich schneller als Backpropagation mit Gradientenabstieg und die Parameterwahl liefert mit den Standardwerten $\eta^+ = 1,2$, $\eta^- = 0,5$, $\Delta_{max} = 50$ und $\Delta_{min} = 10^{-6}$ bei vielen Problemen bereits gute Ergebnisse.

Es wurden alternative Strategien für Gewichtsänderungen und Gewichtsaktualisierungen beim Vorzeichenwechsel der partiellen Ableitung der Fehlerfunktion entwickelt [124], welche die Konvergenzgeschwindigkeit des Verfahrens weiter verbessern können.

Durch eine Adaption der Gewichtsänderungen Δ kann die globale Konvergenz des Algorithms, d. h. die Konvergenz gegen ein (nicht notwendigerweise globales) Optimum von jeder beliebigen Initialisierung der Gewichte aus, garantiert werden [125]. Dabei wird die Schrittweite gemäß der Regel von Wolfe [126, 127] so gewählt, dass die Fehlerfunktion in jedem Schritt verringert wird. Die Umsetzung der Regel erlaubt zusätzlich eine Erhöhung der Konvergenzgeschwindigkeit [125].

7.2.4.5 Simulated annealing-Implementierung (SARprop)

Da Gradientenabstiegs-Algorithmen strikt der stärksten Abstiegsrichtung folgen, konvergieren sie oft gegen lokale Minima, Abbildung 7.4.

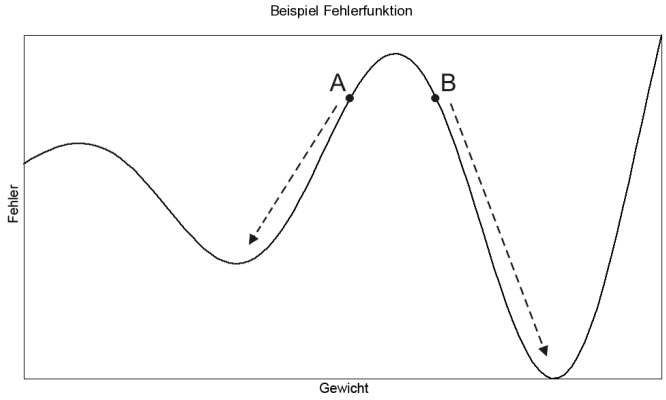


Abbildung 7.4: *Gradientenabstieg-Algorithmen konvergieren für die beiden Ausgangspunkte A und B gegen zwei verschiedene lokale Minima.*

Simulated Annealing-Ansätze [128] versuchen, diese Einschränkung zu umgehen, indem die Umgebung des Gewichtsvektors stochastisch abgetastet wird und auch Schritte zugelassen werden, bei denen sich der Fehler verschlechtert. Dadurch soll das Überwinden lokaler Maxima ermöglicht werden, hinter denen möglicherweise ein besseres Minimum liegt. Eine Verschlechterung von einem Punkt x_α im Iterationsschritt α zu einem Punkt $x_{\alpha+1}$ mit Fehler $E(x_\alpha) < E(x_{\alpha+1})$ wird dabei nur mit einer Wahrscheinlichkeit

$$p = e^{-\frac{E(x_{\alpha+1})-E(x_\alpha)}{T_\alpha}} \tag{7.2.5}$$

zugelassen. Der „Temperaturparameter" T_α wird im Verlauf der Iteration verringert. Dadurch werden anfangs auch starke Verschlechterungen akzeptiert, so dass eine große Umgebung abgesucht und lokale Maxima überwunden werden können. Durch Abnahme von T_α wird es immer unwahrscheinlicher, dass ein lokales Minimum zugunsten eines anderen lokalen Minimums größeren Fehlers verlassen wird. Das Suchgebiet zieht sich zusammen und verlagert sich zu einem Gebiet kleiner Fehlerwerte, bis der Algorithmus schließlich bei sehr kleinen T_α keine Schritte mehr zu Verschlechterungen des Fehlers zulässt und in einen Gradientenabstieg übergeht. Die Konvergenz des Algorithmus stellt Anforderungen an die Verteilungsfunktion, mit der die Umgebung des Gewichtsvektors abgetastet wird. Aus dieser leitet sich die Vorschrift zur Erniedrigung des Temperaturparameters ab [128].

Simulated Annealing bildet den Anordnungsvorgang der Atome bei der Abkühlung eines Kristalls nach. Bei hohen Temperaturen haben die Atome eine hohe Beweglichkeit im Kristall, die es ihnen ermöglicht, sich auch durch energetisch ungünstige Gebiete hindurch zu bewegen. Bei Verringerung der Temperatur steht ihnen immer weniger Energie zur Überwindung von Potentialbarrieren zur Verfügung

und die Atome bewegen sich in das nächstgelegene energetische Minimum. Wenn die Temperatur hinreichend langsam erniedrigt wird, haben die Atome genügend Zeit, ein geordnetes Kristallgitter zu bilden, was der minimalen Gesamtenergie des Systems entspricht. Wenn die Temperatur zu schnell erniedrigt wird, „frieren" Atome auf Zwischengitterplätzen ein und der Zustand niedrigster Energie kann nicht mehr erreicht werden.

Zur Optimierung der Gewichte kann ein Simulated Annealing-Schema direkt auf die Fehlerfunktion angewendet werden. Da das stochastische Absuchen der Fehlerfunktion für hochdimensionale Probleme, in diesem Fall einer großen Anzahl an Gewichten, einen exponentiell steigenden Rechenaufwand zur Folge hat, ist dieses Verfahren außer bei sehr einfachen Netzen ineffektiv [129]. Versuche, die Fehlerfunktion von MLP-Netzen für Datensätze dieser Arbeit direkt mit Simulated Annealing zu optimieren, waren nicht erfolgversprechend, da selbst nach einem Vielfachen der Rechenzeit von Backpropagation-Algorithmen noch keine Ergebnisse erzielt werden konnten, die mit deren Optimierungsfortschritten vergleichbar waren.

SARprop [130] ist eine Simulated Annealing-Implementierung des Rprop-Algorithmus, die aus zwei Erweiterungen besteht.

Die erste Erweiterung besteht darin, dass zu den Gewichten ein Rauschen addiert wird, wenn der Fehlergradient das Vorzeichen wechselt und der Betrag der Gewichtsänderung kleiner ist als eine Schranke proportional zum *Root Mean Square*-Fehler $RMS = \sqrt{MSE}$. Die Stärke des Rauschens ist dabei proportional zum *RMS*-Fehler und dem aktuellen Temperaturparameter T_α. Das Rauschen soll das Verlassen lokaler Minima ermöglichen und die Schranke soll verhindern, dass der zugrundeliegende Rprop-Algorithmus zu stark gestört wird.

Die zweite Erweiterung ist die Addition eines temperaturabhängigen Weight Decay-Terms auf die Fehlerfunktion, so dass sich der modifizierte Fehlergradient ergibt zu

$$\frac{\partial E}{\partial w_{ij}}^{SARProp} = \frac{\partial E}{\partial w_{ij}} - konst \cdot w_{ij} \cdot 2^{-\alpha T_a}. \tag{7.2.6}$$

Durch die Limitierung der Gewichte zu Beginn des Trainings soll ein gleichmäßigeres Absuchen der Fehlerlandschaft ermöglicht werden.

Zu Rprop und SARprop existiert eine Reihe von Erweiterungen, welche die Konvergenzeigenschaften der Algorithmen noch weiter verbessern können [131].

7.2.4.6 Abbruchbedingung für das Lernen

Ziel des Trainings neuronaler Netze ist neben der möglichst exakten Wiedergabe der Lerndaten eine gute Generalisierungsfähigkeit, so dass auch zu im Training unbekannten Eingangsdaten eine treffende

Vorhersage gemacht werden kann. Dies legt die Verwendung eines Trainings- und eines Testdatensatzes nahe. Nach zufälliger Initialisierung der Gewichte nahe Null lernt das Netz auf dem Trainingsfehler. Dabei verringert sich durch die steigende Generalisierungsfähigkeit auch der Testfehler. Die zunehmende Anpassung an die Trainingsdaten führt ab einer bestimmten Epoche zu einem Anstieg des Testfehlers, Abbildung 7.5. Hier beginnt die Überanpassung, bei der sich der Trainingsfehler immer weiter verbessert, bis ein lokales Minimum gefunden wurde. Das Training kann abgebrochen werden, sobald das Minimum des Testfehlers durchlaufen wurde (early stopping). Die Gewichte zur Epoche des niedrigsten Testfehlers werden schließlich ins Modell übernommen.

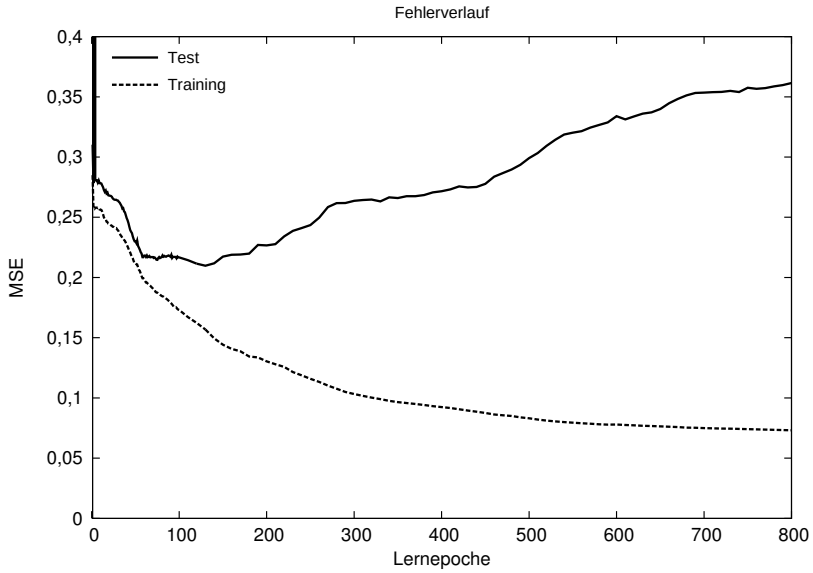


Abbildung 7.5: *Rprop-Fehlerverlauf für ein MLP am Beispiel von HIC-Daten. Der minimale Fehler auf den Testdaten wurde nach 130 Epochen erreicht.*

7.2.5 Anpassung der Netztopologie

In den vorherigen Abschnitten wurden Verfahren gezeigt, mit denen die Gewichte einer fest vorgegebenen Netztopologie trainiert werden können. Wie viele Neuronen und Gewichte ein MLP mindestens zur bestmöglichen Abbildung einer Zielfunktion benötigt, hängt vom Problem ab. Ein MLP kann um beliebig viele Neuronen und Gewichte erweitert werden, ohne dass seine Lösungsfähigkeit dadurch vermindert wird, denn das erweiterte Netz geht durch Nullsetzen aller neu hinzu gefügten Gewichte in das ursprüngliche Netz über. Daher kann allgemein zur Lösung eines Problems immer eine übertrieben komplexe Netztopologie zugrundegelegt werden. Das Lernen erfordert dann einen entsprechend höheren Rechenaufwand. Viel problematischer ist aber, dass die zu optimierende Fehlerfunktion durch Hinzufügen von Neuronen und Gewichten immer mehr zerklüftet. Es entstehen dabei viele lokale Minima, die das Finden

des globalen oder zumindest eines guten lokalen Minimums erschweren. Dadurch wird die Entwicklung von Verfahren motiviert, die nach optimal an das Problem angepassten Netzwerktopologien suchen. Hierzu existieren Ansätze, nach denen unterdimensionierte Netze erweitert oder überdimensionierte Netze vereinfacht werden.

7.2.5.1 Erweiternde Verfahren

Ein Beispiel für ein aufbauend topologieoptimierendes Verfahren ist der Cascade Correlation-Algorithmus [132]. Hierbei wird mit einem Netz ohne verdeckte Ebenen begonnen, das die Neuronen der Eingangsebene und ein Bias-Neuron enthält. Das Netz wird trainiert und anschließend um ein verdecktes Neuron erweitert, dessen Eingabe mit der Ausgabe aller Neurone der Eingabeebene und ggf. bereits existierenden verdeckten Neuronen verbunden wird. Die Eingangsgewichte dieses neuen verdeckten Neurons werden derart angepasst, dass seine Ausgabe betragsmäßig am größten ist, wenn der Fehler des Restnetzes maximal ist. Anschließend wird seine Ausgabe vollständig mit der Ausgabeschicht verbunden und alle Gewichte zu den Ausgabeschichten werden neu trainiert. Die Eingabegewichte zu den verdeckten Neuronen werden im Folgenden nicht mehr verändert. Durch schrittweises Hinzufügen weiterer verdeckter Neuronen kann der Restfehler des Netzes weiter verkleinert werden. Ein Cascade Correlation-Netz hat folglich keine MLP-Struktur.

7.2.5.2 Vereinfachende Verfahren

Überdimensionierte Netze können durch Entfernen (Pruning) von Gewichten oder Neuronen vereinfacht werden. Neben Verfahren wie Weight Decay (Abschnitt 7.2.4.2), die große Gewichte durch Berücksichtigung in der Fehlerfunktion vermeiden, existieren auch Verfahren zur Sensitivitätsbewertung von Verbindungen oder Neuronen. Ein Beispiel hierfür ist der Optimal Brain Damage-Algorithmus [133]. Hierbei wird das Ausgangsnetzwerk zunächst trainiert, bis ein Optimum erreicht ist. Basierend auf einer Taylor-Approximation der Fehlerfunktion wird anschließend für jedes einzelne Gewicht berechnet, wie groß die Änderung der Fehlerfunktion bei Weglassen dieses Gewichts wäre. Gewichte, die nur einen kleinen Einfluss auf die Fehlerfunktion haben, werden dann weggelassen und das vereinfachte Netz wird wieder bis zu einem Optimum trainiert. Falls sich das neue Netz verbessert oder nicht signifikant verschlechtert hat, können weitere Pruning-Schritte durchgeführt werden.

7.2.5.3 Genetische Verfahren

Genetische Algorithmen sind heuristische Verfahren, deren Vorgehen sich an der biologischen Evolution orientiert. Hierbei werden aus einer Menge (Population) von Lösungskandidaten (Individuen) die

am besten geeigneten nach folgendem Schema ausgewählt, verändert und miteinander kombiniert. Jedes Individuum wird als Genom dargestellt, d.h. als Folge von Werten, die seinen Informationsgehalt repräsentieren. Dieses Verfahren wird über so viele Schritte (Generationen) wiederholt, bis die Lösung sich nicht weiter verbessert oder hinreichend gut ist.

- Initialisierung: Eine Ausgangspopulation wird zufällig oder unter Berücksichtigung von problemspezifischem Vorwissen erstellt.

- Evaluation: Eine Fitnessfunktion ermittelt die Güte jedes einzelnen Individuums der aktuellen Generation.

- Selektion: Als Basis für die nächste Generation werden Individuen zufällig ausgewählt, wobei solche mit besserer Fitness mit höherer Wahrscheinlichkeit gewählt werden.

- Mutation: Teile der Genome einzelner selektierter Individuen werden zufällig geändert.

- Rekombination (Crossover): zufällig gewählte einander entsprechende Teilstücke von Genomen zweier Individuen (Eltern) werden untereinander vertauscht und zu einem neuen Individuum (Kind) kombiniert.

- Erstellen der Folgegeneration aus mutierten und rekombinierten Individuen sowie, je nach Verfahren, auch Individuen aus der aktuellen Generation.

Voraussetzung für die Anwendung genetischer Algorithmen ist die Darstellbarkeit der Lösungskandidaten als Genome, so dass Teile davon mutiert und rekombiniert werden können. Die Topologien der Netze können dazu z. B. durch Bitfolgen repräsentiert werden, aus denen Teile mutiert oder rekombiniert werden können. Weiterhin muss eine Fitnessfunktion existieren, mit der die einzelnen Individuen verglichen werden können. Weitere Anforderungen wie etwa Stetigkeit oder Differenzierbarkeit werden nicht an die Fitnessfunktion gestellt. Das ermöglicht genetischen Algorithmen die Bearbeitungen von Problemen, die mit gradientenbasierten Verfahren nicht lösbar sind. Durch die Auswahl besser geeigneter Individuen können Optima der Fehlerfunktion gefunden werden, während die zufälligen Variationen der Population das Verlassen lokaler Minima ermöglichen.

Genetische Algorithmen können sowohl zur Optimierung der Gewichte eines neuronalen Netzes, als auch zur Optimierung der Netztopologie benutzt werden [29]. Die zweckmäßige Darstellung der Netztopologie als Genom ist dabei eine weitere Herausforderung [134]. Bei einer Topologieoptimierung beinhaltet die Bewertung der Fitness jedes Individuums eine Optimierung seiner Gewichte. Da es im Allgemeinen nicht garantiert werden kann, dass das Training der Gewichte immer das globale Optimum findet, wird die Fitnessfunktion verrauscht. Genetische Algorithmen haben sich aber in vielen Fällen als

relativ robust gegenüber verrauschten Problemen erwiesen [134]. Die Gewichtsoptimierungen innerhalb jeder Generation führen zu einem hohen Rechenaufwand. Da die Individuen aber unabhängig voneinander ausgewertet werden können, sind genetische Algorithmen gut parallelisierbar. Weiterhin können die Trainingsdauern durch Vererbung der Gewichte der Eltern auf die Kinder gegenüber zufälliger Neuinitialisierungen beschleunigt werden [135]. Es existieren auch Strategien zur gleichzeitigen Optimierung von Topologie und Gewichten, wodurch die Geschwindigkeit der Evolution gesteigert werden kann [136].

7.3 Klassifikatoren und Bayessche-Netze

Klassifikatoren teilen Objekte in Klassen ein. Bei kontinuierlichen Zielgrößen wird dazu der Wertebereich in Intervalle (Klassen) unterteilt, denen die Eingangsvektoren zugeordnet werden. Eine Funktionsapproximation wird ermöglicht, indem die Definitionsbereiche der Eingangsgrößen komponentenweise in Intervalle unterteilt werden. Die Wahl dieser Einteilung muss dem Problem angepasst sein. Zur Vorhersage für neue Eingabevektoren wird die Zielklasse ermittelt, die den Eingabeintervallen zugeordnet ist, in denen der Eingabevektor liegt. Ein solches Klassifikationsverfahren kann demzufolge beliebig unstetige Zusammenhänge abbilden, wobei jedoch keine stetige Inter- oder Extrapolation möglich ist wie beispielsweise bei der Approximation durch Polynome oder neuronale Netze. Vorhersagen des Änderungsverhaltens der Zielfunktion bei kleinen Änderungen des Eingangsvektors sind mit diesen Verfahren nicht möglich, daher werden sie auch nicht zur Optimierung verwendet. Da diese Verfahren bei der Anpassung an die Daten nicht durch die Bedingung stetiger bzw. stetig differenzierbarer Übergänge zwischen einzelnen Trainingsmustern eingeschränkt werden, sind auch mit verhältnismäßig wenigen Modellparametern gute Anpassungen an Probleme hoher Komplexität möglich. Die Anzahl der Modellparameter ist dabei die Anzahl der Intervalle. Die Anpassungsgüte eines passend gewählten Klassifikators kann damit als Maß für die Modellierbarkeit des Problems betrachtet werden. Probleme, bei denen die zu gleichen oder ähnlichen Trainingsvektoren gemessene Zielgröße stark streut, lassen sich dann auch mit ideal angepassten Klassifikatoren nur mit einem Bestimmtheitsmaß $R^2 < 1$ abbilden.

Bayessche Netze [137] bestehen aus Knoten, die durch Kanten miteinander verbunden sind. Jeder Knoten repräsentiert eine Zufallsvariable, die bestimmte Eigenschaften, z. B. Zahlenwerte, annehmen kann. Diesen werden absolute oder durch andere Knoten bedingte Wahrscheinlichkeitsverteilungen zugeordnet. Die Kanten sind kausal gerichtet und stellen die direkten Abhängigkeiten von Eltern-Knoten (Ursache) zu Kinder-Knoten (Wirkung) dar. Zyklen, durch die irgendein Knoten über andere Knoten wieder auf sich selbst wirkt, sind nicht erlaubt. Es wird jedoch keine Baumstruktur vorausgesetzt, d. h. es sind auch mehrere verschiedene Wege von einem Knoten zu einem anderen erlaubt.

Eine Funktionsapproximation der Zielgröße y als Funktion des Vektors x, $y = f(x_1, \ldots, x_m)$, stellt

sich in wahrscheinlichkeitstheoretischer Sicht dar als die Ermittlung des wahrscheinlichsten Ereignisses des Zielintervalls y unter der Bedingung, dass ein bestimmter Vektor x^α der Eingangsgrößen gemessen wurde:

$$P(y|x^\alpha). \tag{7.3.1}$$

Die bedingte Wahrscheinlichkeit gibt an, mit welcher Wahrscheinlichkeit Ereignis A eintritt, wenn Ereignis B bereits eingetreten ist. Es gilt

$$P(A|B) = \frac{P(A \cap B)}{P(B)}, \tag{7.3.2}$$

wobei $P(A \cap B)$ die a priori-Wahrscheinlichkeit des gleichzeitigen Eintretens der Ereignisse A und B bezeichnet und $P(B)$ die a priori-Wahrscheinlichkeit des Ereignisses B. Unter a priori-Wahrscheinlichkeit versteht man die Anfangswahrscheinlichkeit für ein Ereignis, d. h. ohne vorherige Festlegung der Größen, von denen das Ereignis abhängt.

Bayessche Netze können somit die wahrscheinlichste Ausgabe zu einer Netzeingabe berechnen, indem die Werte derjenigen Knoten vorgegeben werden, welche die Eingangsvariablen repräsentieren. Es können aber selbst dann Ausgabewerte berechnet werden, wenn nicht zu allen Eingangsvariablen Werte vorliegen. In diesem Fall ermittelt das Netz die wahrscheinlichste Ausgabe im Rahmen der aus den Trainingsdaten gelernten Wahrscheinlichkeiten.

Beim Trainieren von Bayesschen Netzen vorgegebener Netztopologie wird nach den bedingten Wahrscheinlichkeitsverteilungen der Knoten gesucht, welche die Trainingsdaten am genauesten wiedergeben. Als weiterer Aspekt des Trainings kann die Anpassung der Netztopologie an das Problem vorgenommen werden.

7.3.1 Naive Bayes-Klassifikator

Damit Klassifikatoren kontinuierliche Werte verarbeiten können, müssen diese zunächst diskretisiert werden, indem Intervalle vorgegeben werden und alle Werte, die in einem Intervall liegen, als Objekte der selben Klasse betrachtet werden. Eine angemessene Diskretisierung der Eingangs- und Zielgrößen ist Teil des Klassifizierungsproblems bei Funktionsapproximationen [138]. Der Naive Bayes-Klassifikator kann als Spezialfall eines sternförmigen Bayes-Netzes aufgefasst werden, das die Vektoren der Eingangsvariablen einer Klasse der Zielgröße zuordnet.

Bayes-Klassifikatoren basieren auf dem Bayes-Theorem, das für diskrete Variablen folgende Form annimmt:

$$P(y|x^\alpha) = \frac{P(x^\alpha|y) \cdot P(y)}{P(x^\alpha)} \tag{7.3.3}$$

Danach wird die bedingte *a posteriori*-Wahrscheinlichkeit $P(y|x^\alpha)$, dass eine bestimmte Zielgröße y gemessen wird, nachdem ein bestimmter Eingangsvektor x^α festgelegt wurde, zurückgeführt auf ihre inverse bedingte Wahrscheinlichkeit $P(x^\alpha|y)$. $P(x^\alpha|y)$ bedeutet, dass zu einer Beobachtung y eine bestimmte Eingangsgröße x vorliegt. $P(x^\alpha|y)$ sowie die a priori-Wahrscheinlichkeit der Zielgröße y können durch die relativen Häufigkeiten der klassifizierten Trainingsdaten bestimmt werden. $P(x^\alpha)$ ist unabhängig von y und kann als Konstante im Hinblick auf die spätere Renormierung der Wahrscheinlichkeiten auf 1 ignoriert werden. Der Naive Bayes-Klassifikator macht die Annahme, dass die Eingangsgrößen x_i statistisch unabhängig voneinander sind [139], so dass $P(x^\alpha|y) = \prod_{i=1\ldots m} P(x_i^\alpha|y)$ gilt. Damit lässt sich $P(y|x^\alpha)$ mit geringem Rechenaufwand bestimmen und erlaubt somit auch die Modellierung hochdimensionaler Probleme. Außerdem reduziert diese Annahme die Menge der benötigten Trainingsdaten auf ein realistisches Maß. Die Annahme der Unabhängigkeit ist in der Realität nur selten erfüllt, weshalb die Vereinfachung für den Zähler in Gleichung 7.3.3 unrealistisch sein kann und zu einer Verzerrung der Wahrscheinlichkeiten $P(y|x^\alpha)$ führt. Abbildung 1.2 auf Seite 12 zeigt einen Fall, bei dem zwei Eingangsparameter eine statistische Abhängigkeit in den Messungen zeigen. Da als Ausgabewert aber lediglich der wahrscheinlichste Wert betrachtet wird und folglich die Differenzen nachfolgender Wahrscheinlichkeiten keine Rolle spielt, liefert das Verfahren bei reinen Klassifikationsproblemen oft trotzdem gute Ergebnisse.

Beispiel

Tabelle 7.1 zeigt einen einfachen Beispieldatensatz zur Zugfestigkeit R_m, mit dem die Anwendung des Naive Bayes-Klassifikators veranschaulicht werden soll.

C	Mo	TEK	R_m
hoch	hoch	hoch	mittel
hoch	hoch	niedrig	hoch
niedrig	niedrig	hoch	niedrig
niedrig	niedrig	niedrig	niedrig
niedrig	hoch	niedrig	mittel
niedrig	hoch	hoch	niedrig
hoch	niedrig	niedrig	mittel
hoch	hoch	niedrig	hoch

Tabelle 7.1: *Trainingsdaten für das Naive Bayes-Klassifikator Beispiel*

Hierbei wurden die kontinuierlichen Messwerte der Eingangsparameter C-Gehalt, Mo-Gehalt und Endkühltemperatur TEK den Klassen {hoch, niedrig} und die Zielgröße R_m den Klassen {hoch, mittel, niedrig} zugeordnet.

Als Ausgabe des Naive Bayes-Klassifikators wird für jede Kombination von Eingangsparametern die wahrscheinlichste Klasse von R_m gewählt:

$$R_m = \underset{R_m^j \in \{\text{niedrig, mittel, hoch}\}}{\arg\max} P(R_m^j) \cdot \prod_{x_i} P(x_i | R_m^j)$$

Die Wahrscheinlichkeiten werden mittels der Anzahl der Realisierungen je Klasse abgeschätzt. Für die a-Priori-Wahrscheinlichkeiten von R_m gilt dann bei insgesamt acht Messungen:

$P(R_m^{hoch}) = 2/8,\ P(R_m^{mittel}) = 3/8, P(R_m^{niedrig}) = 3/8$

und für die bedingten Wahrscheinlichkeiten mit $\left|\left\{R_m^{hoch}\right\}\right| = 2$, $\left|\left\{R_m^{mittel}\right\}\right| = 3$ und $\left|\left\{R_m^{niedrig}\right\}\right| = 3$:

$$\begin{array}{lll}
P(C^{hoch}|R_m^{hoch}) = 2/2, & P(C^{hoch}|R_m^{mittel}) = 2/3, & P(C^{hoch}|R_m^{niedrig}) = 0/3 \\
P(C^{niedrig}|R_m^{hoch}) = 0/2, & P(C^{niedrig}|R_m^{mittel}) = 1/3, & P(C^{niedrig}|R_m^{niedrig}) = 3/3 \\
P(Mo^{hoch}|R_m^{hoch}) = 2/2, & P(Mo^{hoch}|R_m^{mittel}) = 2/3, & P(Mo^{hoch}|R_m^{niedrig}) = 1/3 \\
P(Mo^{niedrig}|R_m^{hoch}) = 0/2, & P(Mo^{niedrig}|R_m^{mittel}) = 1/3, & P(Mo^{niedrig}|R_m^{niedrig}) = 2/3 \\
P(TEK^{hoch}|R_m^{hoch}) = 0/2, & P(TEK^{hoch}|R_m^{mittel}) = 1/3, & P(TEK^{hoch}|R_m^{niedrig}) = 2/3 \\
P(TEK^{niedrig}|R_m^{hoch}) = 2/2, & P(TEK^{niedrig}|R_m^{mittel}) = 2/3, & P(TEK^{niedrig}|R_m^{niedrig}) = 1/3
\end{array}$$

Nun soll R_m für den in den Trainingsdaten nicht enthaltenen Fall $C^{hoch}, Mo^{niedrig}, TEK^{hoch}$ abgeschätzt werden. Dazu werden die (nicht normierten) Wahrscheinlichkeiten für alle drei R_m-Klassen berechnet:

$$\begin{aligned}
&P(C^{hoch}, Mo^{niedrig}, TEK^{hoch}|R_m^{hoch}) \\
&= P(R_m^{hoch}) \cdot P(C^{hoch}|R_m^{hoch}) \cdot P(Mo^{niedrig}|R_m^{hoch}) \cdot P(TEK^{hoch}|R_m^{hoch}) = 0
\end{aligned}$$

$$\begin{aligned}
&P(C^{hoch}, Mo^{niedrig}, TEK^{hoch}|R_m^{mittel}) \\
&= P(R_m^{mittel}) \cdot P(C^{hoch}|R_m^{mittel}) \cdot P(Mo^{niedrig}|R_m^{mittel}) \cdot P(TEK^{hoch}|R_m^{mittel}) = 0,0278
\end{aligned}$$

$$\begin{aligned}
&P(C^{hoch}, Mo^{niedrig}, TEK^{hoch}|R_m^{niedrig}) \\
&= P(R_m^{niedrig}) \cdot P(C^{hoch}|R_m^{niedrig}) \cdot P(Mo^{niedrig}|R_m^{niedrig}) \cdot P(TEK^{hoch}|R_m^{niedrig}) = 0
\end{aligned}$$

Die Klasse R_m^{mittel} hat die höchste Wahrscheinlichkeit und wird somit als Ausgabe des Naive Bayes-Klassifikators gewählt.

Kontinuierliche Variablen

Um eine Regression mit kontinuierlichen Variablen durchzuführen, kann anstatt einer reinen Diskretisierung und Kategorisierung der Variablen auch das Bayes-Theorem für kontinuierliche Wahrscheinlichkeitsverteilungen verwendet werden,

$$p(y|x) = \frac{p(x|y)p(y)}{\int p(x|y)p(y)dy}, \tag{7.3.4}$$

wobei p die zu bestimmenden Wahrscheinlichkeitsdichten der Messgrößen bezeichnet. Nach Wahl einer Wahrscheinlichkeitsdichtefunktion zur Modellierung des Problems, z. B. einer Gaußverteilung, wird diese an die Trainingsdaten angepasst und $p(x|y)$ sowie $p(y)$ werden bestimmt. Je nachdem, ob das Modell eine optimale Vorhersage bezüglich *MSE* oder absolutem Fehler treffen soll, wird entweder der Erwartungswert oder der Median von $p(y|x)$ als Modellwert zurückgegeben [140].

Ob sich eine Naive Bayes-Modellierung basierend auf Gleichung 7.3.3 oder auf Gleichung 7.3.4 besser eignet, hängt vom Problem ab. Naive Bayes-Regressionsprobleme, die auf kontinuierlichen Wahrscheinlichkeitsverteilungen basieren, können verglichen mit Standard-Regressionsverfahren zu schlechter Approximationsgüte führen [140]. Grund dafür ist, dass hierbei nicht nur ein einzelner Wert maximaler Wahrscheinlichkeit betrachtet wird, sondern über die Wahrscheinlichkeitsdichte integriert wird, wodurch Fehler in der Verteilung durch die Unabhängigkeitsannahme in das Ergebnis mit einfließen. Diskretisierende Naive Bayes-Verfahren können, abhängig von der Wahl der Verteilungsfunktion bzw. der Diskretisierung, zu besseren Ergebnissen führen als kontinuierliche Ansätze [141].

7.3.2 One-Dependence Estimators

Eine Möglichkeit zur Auflockerung der Einschränkungen, die sich aus der Annahme der Unabhängigkeit der Eingangsgrößen ergeben, ist die Berücksichtigung paarweiser Abhängigkeiten von Komponenten der Eingangsgrößen, so dass sich für die gemeinsame Wahrscheinlichkeit

$$P(x \cap y) = P(y \cap x_i)P(x|y \cap x_i) \tag{7.3.5}$$

ergibt. Die höhere Anzahl der Klassen bei dieser Aufteilung führt allerdings zu einer Ausdünnung der Häufigkeiten pro Klasse, wodurch sich die Varianz solcher Modelle vergrößert. Zur Verringerung der Varianz mittelt der Averaged One-Dependence Estimator (AODE) deshalb über mehrere ausgewählte Einzelmodelle, die jeweils paarweise Abhängigkeiten bezüglich einer Komponente x_k berücksichtigen, falls die Klassen bezüglich $x_i x_k \forall i$ genügend Werte für eine vertrauenswürdige Abschätzung der Wahrscheinlichkeiten enthält [142]. Bei der Erweiterung Weightily Averaged One-Depende Estimators (WAODE) werden diese paarweisen Abhängigkeiten zusätzlich gewichtet [143].

Glossar

Backpropagation

Ebenenweise Rückpropagation der Fehler der einzelnen Gewichte eines neuronalen Netzes

Backward Elimination (BE)

Rückwärtselimination, bei der ein Modell schrittweise vereinfacht wird

Biasfehler

Abbildungsfehler, der durch ein zu einfaches (starres) Modell entsteht

Capped Data

Deckelung der Messwerte durch einen Schwellwert

Diskretisierer

Algorithmus, der kontinuierliche Zahlenwerte (Messwerte) problemspezifisch in Klassen einteilt

Eingangsgrößen, Eingangsparameter

Unabhängige Variablen eines Modells, hier: Prozessparameter

Forward Selection (FS)

Vorwärtsselektion, bei der ein Modell schrittweise erweitert wird

Klasse

Intervall, das ähnliche Elemente enthält

Kreuzvalidierung

Aufeinanderfolgende disjunkte Aufteilung der Messdaten in Test- und Trainingsdaten, so dass über alle Kreuzvalidierungsschritte nacheinander alle Daten dem Modell als Test- und Trainingsdaten präsentiert werden

Least Squares-Anpassung

Anpassung der Regressionskoeffizienten eines linearen Polynommodells mit minimalem quadratischen Fehler

Lernepoche

Lernschritt beim Training neuronaler Netze, bei dem alle Trainingsdaten durch das Netz propagiert werden und anschließend eine Fehlerkorrektur ermittelt wird

Lernparameter

Parameter, die die Modellanpassung steuern

Multi-Layer Perceptron (MLP)

Neuronales Netz, bei dem die Neuronen in aufeinanderfolgenden Ebenen angeordnet sind und die Verbindungen (Gewichte) jeweils von den Neuronen einer Ebene zur denen in der Folgeebene verlaufen

Polynomstruktur

Menge der im Polynom enthaltenen Eingangsgrößen und Produkten daraus

Prozessparameter

Variablen, die den physikalisch-metallurgischen Prozess beschreiben

Rauschen

Alle statistischen Schwankungen, die zu einer Abweichung zwischen Messwert und realem Zusammenhang führen

Regressionsparameter

Anpassungskoeffizienten eines linearen Regressionsmodells

Shrinkage

Abnahme des Bestimmtheitsmaßes auf den Testdaten gegenüber den Trainingsdaten

Testdaten

Datensatz, der während des Lernens eines Modells auf den Trainingsdaten zur Abschätzung dessen Generalisierungsfähigkeit verwendet wird

Topologieoptimierung

Anpassung der Anzahl Neuronen und Verbindungen eines neuronalen Netzes an die Komplexität des Problems

Trainingsdaten

Datensatz zur Anpassung der Koeffizienten (Regressionsmodell) bzw. Gewichte (neuronales Netz)

Validierungsdaten

Ein zu den Test- und Trainingsdaten disjunkter Datensatz, der erst dem finalen Modell zur Abschätzung der Generalisierungs- oder Extrapolationsfähigkeit präsentiert wird

Varianzfehler

Abbildungsfehler, der durch ein zu komplexes (oszillierendes) Modell entsteht

Weightily Averaged one-dependence estimators (WAODE)

Klassifikator, der auf bedingten Wahrscheinlichkeiten basiert und paarweise Abhängigkeiten der Eingangsgrößen gewichtet berücksichtigt

Zielgröße

Abhängige Variable eines Modells, hier: die zu modellierende Messgröße

Symbolverzeichnis

x	Eingangsparameter
y	gemessene Zielgröße
n	Anzahl Messungen
m	Anzahl Eingangsparameter
p	Anzahl kombinierter Terme Polynommodell bzw. Verbindungen neuronales Netz
q	maximale Ordnung der Auswahlterme für adaptive Polynommodelle
$\delta^{\pm}$	Anzahl pro Anpassungsschritt hinzugefügter (+) bzw. entfernter (-) Terme oder Gewichte
n_{cap}	Anzahl Capped Data-Iterationsschritte
A	Matrix der Regressionsvariablen
a	Vektor der Regressionsparameter
D	Datensatz zum Anpassen und Testen der Modelle
D_{train}	Teilmenge zum Anpassen der Modelle
D_{test}	Teilmenge zum Test der Anpassungsgüte
R^2	Bestimmtheitsmaß bezüglich eines Datensatzes: Index $train$ = Trainingsdaten, $lern$ = Lerndaten, $test$ = Testdaten, $valid$ = Validierungsdaten
MSE	mittlerer quadratischer Fehler
σ	Streuung
RSS	Residuenquadratsumme

AIC	Akaike Informationskriterium
BIC	Bayes Informationskriterium
C_p	Mallows Informationskriterium
Ein_i	Eingabe zum Neuron i
Aus_i	Ausgabe des Neurons i

Literaturverzeichnis

[1] R. J. Fruehan. *The Making, Shaping and Treating of Steel (Steel Making and Refining)*. AISE Steel Foundation, 1998.

[2] Y. Ueshima, S. Mizoguchi, T. Matsumiya, and H. Kajioka. Analysis of solute distribution in dendrites of carbon steel with delta/gamma transformation during solidification. *Metallurgical and Materials Transactions B*, 17:845–859, 1986. 10.1007/BF02657148.

[3] A. Ghosh. Segregation in cast products. *Sadhana*, 26:5–24, 2001. 10.1007/BF02728476.

[4] T. Gladman. Precipitation hardening in metals. *Materials Science and Technology*, 15:30–36, 1999.

[5] R. Dekkers. *Non-metallic inclusions in liquid steel*. PhD thesis, Katholieke Universiteit Leuven, 2002.

[6] A. Lederer. Metallurgische Aspekte des thermomechanischen Walzens. *Bänder Bleche Rohre*, 5:117–120, 1982.

[7] H.-J. Kirsch, P. Flüß, W. Schütz, and A. Streißelberger. Neue Eigenschaftskombinationen an Grobblech durch den beschleunigten Kühlprozeß. *Stahl und Eisen*, 119:57–65, 1999.

[8] Verein Deutscher Eisenhüttenleute, editor. *Stahlfibel*. Verlag Stahleisen GmbH, 2002.

[9] B. Verlinden, J. Driver, I. Samajdar, and R. D. Doherty. *Thermo-Mechanical Processing of Metallic Materials*. Elsevier Science, 2007.

[10] W. Jäniche, W. Dahl, H. F. Klärner, W. Pitsch, D. Schauwinhold, W. Schlüter, and H. Schmitz. *Werkstoffkunde Stahl, Band 1: Grundlagen*. Verein deutscher Eisenhüttenleute, 1984.

[11] E. Moeller. *Handbuch Konstruktionswerkstoffe: Auswahl, Eigenschaften, Anwendung*. Hanser, 2007.

[12] Yoshisato Kimura and David P. Pope. Ductility and toughness in intermetallics. *Intermetallics*, 6:567–571, 1998.

[13] D. Gaskell. *Introduction to the Thermodynamics of Materials*. Taylor & Francis, 2003.

[14] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2006.

[15] W. R. Irving. *Continuous Casting of Steel*. The Institute of Materials, 1993.

[16] Y. Bar-Yam. *Dynamics of complex systems*. Perseus Books, Cambridge, MA, USA, 1997.

[17] C. Johnson. 2nd Workshop on Complexity in Design and Engineering. Technical report, University of Glasgow, 2005.

[18] Complexity in Engineering, 2010, COMPENG '10. Feb. 2010.

[19] K. H. Tacke. Cavity Sequences in Continuously Cast Billets: Part I. Analysis of Empirical Data. *Metallurgical and Materials Transactions B*, 30B:751–761, 1999.

[20] K. H. Tacke. Cavity Sequences in Continuously Cast Billets: Part II. Stochastic Models. *Metallurgical and Materials Transactions B*, 30B:763–772, 1999.

[21] K. H. Tacke and M. Hecht. Depth of oscillation marks in thick slab casting. *2nd CSM - VDEh-Seminar on Metallurgical Fundamentals*, pages 260–267, 2007.

[22] K. H. Tacke. Oscillation and surface mark patterns in slab casting. In *Proceedings of the 4th International Conference on Continuous Casting of Steel in Developing Countries, Journal of Iron and Steel Research Int. Vol. 15*, 2008.

[23] H. Pruscha. *Statistisches Methodenbuch: Verfahren, Fallstudien, Programmcodes*. Springer, 2005.

[24] P. Deuflhard and A. Hohmann. *Numerische Mathematik I*. Walter de Gruyter, 1993.

[25] H. Motulsky. The GraphPad Guide to Nonlinear Regression. Technical report, Graphpad Software, 1996.

[26] W. S. Cleveland. Robust Locally Weighted Regression and Smoothing Scatterplots. *Journal of the American Statistical Association*, 74:829–836, 1979.

[27] J. Fan and I. Gijbels. *Local Polynomial Modelling and Its Applications: Monographs on Statistics and Applied Probability*. Chapman & Hall, 1 edition, March 1996.

[28] D. M. Skapura J. A. Freeman. *Neural Networks: Algorithms, Applications, and Programming Techniques*. Addison Wesley, 1992.

[29] R. Rojas. *Neural Networks, A Systematic Introduction*. Springer, 1996.

[30] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Academic Press, 2005.

[31] EN ISO 148-1:2011-01, 2011.

[32] A. Cosham, D. Jones, R. Eiber, and P. Hopkins. Don't drop the drop-weight tear test. *Journal of Pipeline Engineering*, 9:69–84, 2010.

[33] L. Issler, H. Ruoß, and P. Häfele. *Festigkeitslehre - Grundlagen*. Springer, 2003.

[34] R. Adnan, M. N. Mohamad, and H. Setan. Multiple Outliers Detection Procedures in Linear Regression. *Matematika*, 19:29–45, 2003.

[35] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft. When Is "Nearest Neighbor"Meaningful? In *In International Conference on Database Theory*, pages 217–235, 1999.

[36] R. E. Bellman. *Adaptive control processes*. Princeton University Press, 1961.

[37] S. Dalbir, S. S. Kumar, D. I. Jit, Y. S. Bansal, and P. A. Naresh. Polynomial regression model to estimate time since death in adults from rectal temperature in Chandigarh zone of northwest India. *Journal of Indian Academy of Forensic Medicine*, 27:170–175, 2005.

[38] A. C. Changa and J.-P. Hsu. A polynomial regression model for the response of various accelerating techniques on maize wine maturation. *Food Chemistry*, 94:603–607, 2006.

[39] T. R. M. De Beer, W. R. G. Baeyens, Y. van der Heyden, J. P. Remon, C. Vervaet, and F. Verpoort. Influence of particle size on the quantitative determination of salicylic acid in a pharmaceutical ointment using FT-Raman spectroscopy. *European Journal of Pharmaceutical Sciences*, 30(3-4):229–235, 2007.

[40] F. T. Z.-H. Tang, Q.-Z. Yan, and Y. Yu. Polynomial regression calculation of the Earth's position based on millisecond pulsar timing. *Research in Astronomy and Astrophysics*, 12, 2012.

[41] L. Buse, M. Ganea, and Lect. D. Cîrciumaru. Using Linear Regression In The Analysis Of Financial-Economic Performances. *Annals of University of Craiova - Economic Sciences Series*, 2(38):12, May 2010.

[42] M. C. Acock and Y. A. Pachepsky. Estimating Missing Weather Data for Agricultural Simulations Using Group Method of Data Handling. *Journal of applied Meteorology*, 39:1176–1184, 1999.

[43] A. von Eye and C. Schuster. *Regression Analysis for Social Sciences*. Academic Press, 1998.

[44] P. Bühlmann, B. Yu, Y. Singer, and L. Wasserman. Sparse Boosting. *Journal of Machine Learning Research*, 7:1001–1024, 2006.

[45] T. Cleff. *Deskriptive Statistik und moderne Datenanalyse: Eine computergestützte Einführung mit Excel, SPSS und stata*. Gabler Verlag, 2008.

[46] D. M. Allen. The relationship between variable selection and data augmentation and a method of prediction. *Technometrics*, 16:125–127, 1974.

[47] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society Series B Methodological*, 36(2):111–147, 1974.

[48] I. Rivals and L. Personnaz. On Cross Validation for Model Selection. *Neural Computation*, 11(4):863–870, May 1999.

[49] B. Clarke, E. Fokoue, and H. H. Zhang. *Principles and Theory for Data Mining and Machine Learning*. Springer, 2009.

[50] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the 14th international joint conference on Artificial intelligence - Volume 2*, IJCAI'95, pages 1137–1143, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

[51] S. Arlot and A. Celisse. A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4:40–79, 2010.

[52] D. E. Farrar and R. R. Glauber. Multicollinearity in Regression Analysis: The Problem Revisited. *The Review of Economics and Statistics*, 49:92–107, 1967.

[53] N. Brauner and M. Shacham. Considering Error Propagation in Stepwise Polynomial Regression. *Industrial & Engineering Chemistry Research*, 38:4477–4485, 1999.

[54] E. Wendler-Kalsch. Grundlagen und Mechanismen der Wasserstoff-induzierten Korrosion metallischer Werkstoffe. In D. Kuron, editor, *Wasserstoff und Korrosion*, pages 7 – 53. Verlag Irene Kuron Bonn, 2000.

[55] T. Siewert. Analysis of the Catastrophic Rupture of a Pressure Vessel. http://nvlpubs.nist.gov/nistpubs/sp958-lide/350-352.pdf. Abgerufen am 2. 6. 2011.

[56] T. Staudt. *Untersuchungen zur Beständigkeit mikrolegierter, ferritischer Feinkornbaustähle gegen korrosive Schädigung*. PhD thesis, Technische Universität Berlin, 2011.

[57] S. Stigler. Fisher and the 5 *Chance*, 21:12–12, 2008.

[58] Jan M. Hoem. The reporting of statistical significance in scientific journals. *Demographic Research*, 18(15):437–442, June 2008.

[59] R. G. Lomax and D. L. Hahs-Vaughn. *Statistical Concepts: A Second Course*. Routledge Academic, 2012.

[60] C. L. Mallows. Some comments on Cp. *Technometrics*, 15:661–675, 1973.

[61] C. L. Mallows. More comments on Cp. *Technometrics*, 37(4):362–372, November 1995.

[62] Robert W. Kennard. A Note on the Cp Statistic. *Technometrics*, 13(4):899–900, 1971.

[63] J. Shao. Linear Model Selection by Cross-Validation. *Journal of the American Statistical Association*, 88(422):486–494, 1993.

[64] H. Akaike. Information theory and an extension of the maximum likelihood principle. *Second International Symposium on Information Theory*, 1:267–281, 1973.

[65] E. I. George and D. P. Foster. Calibration and Empirical Bayes Variable Selection. *Biometrika*, 87:731–747, 1997.

[66] A. F. Young, J. A. Montoya, C. Sanloup, M. Lazzeri, E. Gregoryanz, and S. Scandolo. AIC and BIC: Comparisons of Assumptions and Performance. November 2005.

[67] C. Rao. Linear model selection by cross-validation. *Journal of Statistical Planning and Inference*, 128(1):231–240, January 2005.

[68] L. Breiman. The Little Bootstrap and Other Methods for Dimensionality Selection in Regression: X-Fixed Prediction Error. 87(419):738–754, 1992.

[69] B. Efron and R. Tibshirani. Cross-validation and the bootstrap: Estimating the error rate of a prediction rule. Technical Report 176, Division of Biostatistics, Stanford University, Stanford, California, May 1995.

[70] B. Efron. *The Jackknife, the bootstrap and other resampling plans*. Number 38 in Regional Conference Series in applied mathematics. Society for Industrial and applied mathematics, Philadelphia, Pa., 1982.

[71] Bradley Efron. Estimating the Error Rate of a Prediction Rule: Improvement on Cross-Validation. *Journal of the American Statistical Association*, 78(382):316–331, 1983.

[72] D. P. Foster and E. I. George. The Risk Inflation Criterion for Multiple Regression. *The Annals of Statistics*, 22(4):1947–1975, 1994.

[73] R. Tibshirani and K. Knight. The Covariance Inflation Criterion for Adaptive Model Selection. *Journal of the Royal Statistical Society B*, 55:757–796, 1999.

[74] J. Ye. On Measuring and Correcting the Effects of Data Mining and Model Selection. *Journal of the American Statistical Association*, 93(441):120–131, 1998.

[75] E. I. George. The Variable Selection Problem. *Journal of the American Statistical Association*, 95(452):1304–1308, December 2000.

[76] W. Zucchini. An Introduction to Model Selection. *Journal of Mathematical Psychology*, 44:41–61, 2000.

[77] R. Rao and Y. Wu. A strongly consistent procedure for model selection in a regression problem. *Biometrika*, 76(2):369–374, 1989.

[78] H. Leeb and B. M. Pötscher. Model selection and inference: facts and fiction. *Econometric Theory*, 21:21–59, 2005.

[79] D. W. Salt, S. Ajmani, R. Crichton, and D. J. Livingstone. An improved approximation to the estimation of the critical F values in best subset regression. *Journal of Chemical Information and Modeling*, 47:143–149, 2007.

[80] Clifford M. Hurvich and Chih-Ling Tsai. A crossvalidatory AIC for hard wavelet thresholding in spatially adaptive function estimation. *Biometrika*, 85(3):701–710, 1998.

[81] C. Z. Wei. On predictive least squares principles. *Annals of Statistics*, 20(1):1–42, 1992.

[82] J. Shao. An asymptotic theory for linear model selection. *Statistica Sinica*, 7(2), 1997.

[83] X. Zheng and W.-Y. Loh. A consistent variable selection criterion for linear models with high-dimensional covariates. *Statistica Sinica*, 7(2):311–325, 1997.

[84] P. T. Pope and J. T. Webster. The Use of an F-Statistic in Stepwise Regression Procedures. *Technometrics*, 14:327–340, 1972.

[85] A. R. Barron, A. Cohen, W. Dahmen, and R. A. DeVore. Approximation and learning by greedy algorithms. *Annals of Statistics*, 36:64–94, 2008.

[86] B. C. Wallet, D. J. Marchette, J. L. Solka, and E. J. Wegman. A Genetic Algorithm for Best Subset Selection in Linear Regression. *Proceedings of the 28th Symposium on the Interface*, 1996.

[87] M. A. Efroymson. Multiple regression analysis. *Mathematical Methods for Digital Computers*, pages 191–203, 1960.

[88] A. J. Miller. Selection of Subsets of Regression Variables. *Journal of the Royal Statistical Society*, 147:389–425, 1984.

[89] L. Wilkinson and G. E. Dallal. Tests of Significance in Forward Selection Regression with an F-to-Enter Stopping Rule. *Technometrics*, 23:377–380, 1981.

[90] H. Wang. Forward Regression for Ultra-High Dimensional Variable Screening. *Journal of the American Statistical Association*, 104(488):1512–1524, 2009.

[91] N. Mantel. Why stepdown procedures in variable selection? *Technometrics*, 12:621–625, 1970.

[92] A. J. Miller. *Subset selection in regression, second edition*. Chapman and Hall/CRC, 2002.

[93] R. A. Sundberg. *Shrinkage Regression*. John Wiley & Sons, Ltd, 2006.

[94] J. B. Copas. Regression, Prediction and Shrinkage. *Journal of the Royal Statistical Society. Series B (Methodological)*, 45(3):311–354, 1983.

[95] A. E. Hoerl and R. W. Kennard. Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12:55–67, 1970.

[96] F. S. M. Batah and S. D. Gore. Ridge Regression Estimator: Combining unbiased and ordinary Ridge Regression methods of estimation. *Surveys in Mathematics and its Applications*, 4:99–109, 2009.

[97] R. Tibshirani. Regression Shrinkage and Selection Via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288, 1994.

[98] H. Zou and T. Hastie. Regularization and variable selection via the Elastic Net. *Journal of the Royal Statistical Society, Series B*, 67:301–320, 2005.

[99] Ching-Kang Ing and Tze Leung Lai. A stepwise regression method and consistent model selection of high-dimensional sparse linear models. *Statistica Sinica*, 21:1473–1513, 2011.

[100] L. Breiman. Better subset regression using the nonnegative garrote. *Technometrics*, 37(4):373–384, November 1995.

[101] P. Bühlmann and B. Yu. Boosting with the L2 Loss: Regression and Classification. *Journal of the American Statistical Association*, 98:324–339, 2003.

[102] P. Bühlmann and E. Zürich. Boosting for high-dimensional linear models. *The Annals of Statistics*, 2006.

[103] C. Leng, Y. Lin, and G. Wahba. A note on the LASSO and related procedures in model selection. Technical report, Statistica Sinica, 2004.

[104] R. W. Lutz and P. Bühlmann. Boosting for high-multivariate responses in high-dimensional linear regression. *Statistica Sinica*, 16(2):471–494, 2006.

[105] R. Lutz, R. Werner, and P. Bühlmann. Conjugate Direction Boosting. *Journal of Computational and Graphical Statistics*, 15(2):287–311, 2006.

[106] Jian Huang, Shuangge Ma, Hongzhe Li, and Cun-Hui Zhang. The Sparse Laplacian Shrinkage Estimator for High-Dimensional Regression. *Annals of Statistics*, 39(4):2021–2046, 2011.

[107] P. Royston and W. Sauerbrei. *Multivariable Model-building*. Wiley, 2009.

[108] P. Burman. A Comparative Study of Ordinary Cross-Validation, v-Fold Cross-Validation and the Repeated Learning-Testing Methods. *Biometrika*, 76:503–514, 1989.

[109] G. H. Golub and C. F. Van Loan. *Matrix computations*. The Johns Hopkins University Press, 1996.

[110] Y. Yang. Regression with Multiple Candidate Models: Selecting or Mixing? *Statistica Sinica*, 13:783–809, 1999.

[111] A Modeling Language for Mathematical Programming. Abgerufen am 1. 7. 2009.

[112] KNIME - Konstanz Information Miner. http://www.knime.org. Abgerufen am 30. 10. 2010.

[113] WEKA Data Mining Software. http://www.cs.waikato.ac.nz/ml/weka/. Abgerufen am 30. 10. 2010.

[114] L. Kurgan, K. Cios, L. A. Kurgan, K. J. Cios, and Senior Member. CAIM Discretization Algorithm. *IEEE Transactions on Knowledge and Data Engineering*, 16:145–153, 2004.

[115] J. G. F. Francis. The QR Transformation A Unitary Analogue to the LR Transformation Part 1. *The Computer Journal*, 4(3):265–271, 1961.

[116] R. Davies. newmat11 numerical library. http://www.robertnz.net/. Abgerufen am 20. 2. 2007.

[117] A. J. Morris and E. B. Martin. Neural Networks - Panacea or Pragmatic Solution. *Proceedings ECSC-Workshop Application of artificial neural network systems in the steel industry*, 1998.

[118] R. P. Lippmann. An Introduction to Computing with Neural Nets. *IEEE ASSP Magazine*, 1987.

[119] G. Cybenko. Approximation by Superpositions of a Sigmodial Function. *Mathematics of Control Signals and Systems*, 2:303–314, 1989.

[120] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning Internal Representations by Error Propagation. In D. E. Rumelhart, J. L. McClelland, et al., editors, *Parallel Distributed Processing: Volume 1: Foundations*, pages 318–362. MIT Press, Cambridge, 1987.

[121] S. E. Fahlman. An Empirical Study of Learning Speed in Back-Propagation Networks. Technical report, 1988.

[122] P. Werbos. Backpropagation: Past and future. In *Proceedings of the IEEE International Conference on Neural Networks*, pages 343–353, 1988.

[123] M. Riedmiller and H. Braun. A direct adaptive method for faster backpropagation learning: the RPROP algorithm. *Proceedings of the IEEE International Conference on Neural Networks (ICNN)*, pages 586–591, 1993.

[124] C. Igel and M. Husken. Empirical evaluation of the improved Rprop learning algorithms. *Neurocomputing*, 50:105–123, 2003.

[125] A. D. Anastasiadis, G. D. Magoulas, and M. N. Vrahatis. New globally convergent training scheme based on the resilient propagation algorithm. *Neurocomputing*, 64:253–270, 2005.

[126] P. Wolfe. Convergence conditions for ascent methods. *SIAM Rev*, 11:226–235, 1969.

[127] P. Wolfe. Convergence conditions for ascent methods II: some corrections. *SIAM Rev*, 13:185–188, 1971.

[128] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science, New Series No. 4598*, 220:671–680, 1983.

[129] Y. Sun. Adaptive Simulated Annealing: An Alternative Approach for the Error Minimization of Neural Networks. Master's thesis, The University of Tennessee, Knoxville, 2001.

[130] N. K. Treadgold and T. D. Gedeon. Simulated Annealing and Weight Decay in Adaptive Learning: The SARPROP Algorithm. *IEEE Tr. Neural Networks*, 9, 4:662–668, 1998.

[131] A. D. Anastasiadis. *Neural networks training and applications using biological data*. PhD thesis, University of London, 2005.

[132] S. E. Fahlman and C. Lebiere. *The Cascade-Correlation Learning Architecture*, 1991.

[133] Y. LeCun, J. S. Denker, and S. A. Solla. Optimal Brain Damage. In *Advances in Neural Information Processing Systems*, pages 598–605. Morgan Kaufmann, 1990.

[134] J. Branke. Evolutionary Algorithms for Neural Network Design and Training. In *Proceedings of the First Nordic Workshop on Genetic Algorithms and its Applications*, pages 145–163, 1995.

[135] D. Whitley and C. Bogart. The evolution of connectivity: Pruning neural networks using genetic algorithms. In *Int. Joint Conf. on Neural Networks, V1*, 1990.

[136] H. Braun and J. Weisbrod. Evolving Neural Feedforward Networks. In R. F. Albrecht, C. R. Reeves, and N. C. Steele, editors, *Proceedings of the International Conference on Neural Networks and Genetic Algorithms (Innsbruck, Austria)*, Wien and New York, 1993. Springer.

[137] I. Ben-Gal. *Encyclopedia of Statistics & Reliability*. Wiley & Sons, 2007.

[138] Y. Yang and G. I. Webb. A Comparative Study of Discretization Methods for Naive-Bayes Classifiers. In *The 2002 Pacific Rim Knowledge Acquisition Workshop, Tokyo*, pages 159–173, 2002.

[139] I. Rish. An empirical study of the naive Bayes classifier. In *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence*, 2001.

[140] E. Frank, L. Trigg, G. Holmes, and I. H. Witten. Naive Bayes for Regression. In *Machine Learning*, pages 5–26, 1998.

[141] J. Dougherty, R. Kohavi, and M. Sahami. Supervised and Unsupervised Discretization of Continuous Features. pages 194–202. Morgan Kaufmann, 1995.

[142] G. I. Webb, J. R. Boughton, and Z. Wang. Not so naive Bayes: Aggregating one-dependence estimators. In *Machine Learning*, pages 5–24, 2005.

[143] L. Jiang and H. Zhang. *PRICAI 2006: Trends in Artificial Intelligence*. Springer, 2006.